## → Introduction to Computer Graphics:—

→ The computer graphics is one of the most effective and commonly used way to communicate the processed information to the user.

→ It displays the information in the form of graphics objects such as pictures, charts, graphs and diagrams instead of simple text.

## → Applications of computer Graphics:

→ User Interfaces
→ Plotting of graphics and chart
→ Office automation and desktop publishing
→ Computer-aided drafting and design.
→ Simulation and Animation.
→ Art and commerce
→ Process Control
→ Cartography

## → Classification of CG applications:—

Uses of computer Graphics

- Type of object (Dimensionality)
  - 2D
  - 3D
- Kind of picture
  - Symbolic
  - Realistic
- Type of Interaction
  - Controllable
  - Non controllable
- Role of Picture
  - Use for representation
  - Use as an end product such as drawing
- Pictorial representation
  - Line drawing
  - Black & white image
  - colour image etc.

## 2. Video Display Devices:

→ The primary output device in a graphics system is a video monitor.

→ The operation of most video monitors is based on the standard Cathode-Ray Tube (CRT) design.

### → Cathode-Ray Tube (CRT)

→ A beam of electrons (cathode rays), emitted by an electron gun, passes through focusing and deflection systems that direct the beam toward specified positions on the phosphor-coated screen.

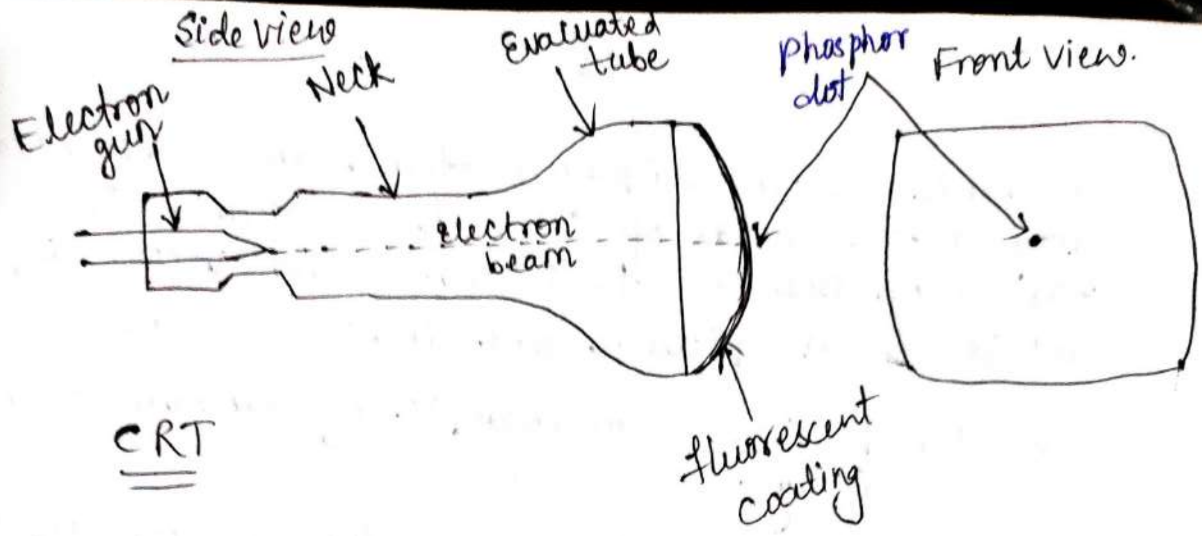
→ The phosphor then emits a small spot of light at each position contacted by the electron beam.

→ Because the light emitted by the phosphor fades very rapidly, some method is needed for maintaining the screen picture.

1. → One way to do this to store the picture information as a charge distribution within the CRT.
   ↳ This charge distribution can then be used to keep the phosphors activated.

2. Redraw the picture repeatedly by quickly directing the electron beam back over the same screen points.
   ↳ This type of display is called a refresh CRT.

→ The frequency at which a picture is redrawn on the screen is referred to as the refresh rate.




CRT

→ Raster Scan Displays :-

Raster Scan Displays are most common type of graphics monitor which employs CRT. It is based on television technology. In raster scan system electron beam sweeps across the screen, from top to bottom ~~Cove~~ covering one row at a time.

→ A memory area called refresh buffer or frame buffer stores picture definition. This memory area called refresh buffer or frame buffer stores picture definition.

→ In raster scan systems refreshing is done at a rate of 60-80 frames/sec.

Scan-line : Each row is referred to as a scan-line.

Pixel or pel : Each screen spot that can be illuminated by the electron beam is called as a pixel.

Aspect Ratio :- the no. of pixel columns divided by the no. of scan lines that can be displayed by the systems.

> smallest unit of a digital image or graphic -

## → Random - Scan Displays :-

In random-Scan Display electron beam is directed only to the areas of screen where a picture has to be drawn. It is also called vector display, as it draws picture one line ~~tt~~ at time.

→ Pen-plotter is an example of random-scan displays.

→ The number of lines regulates refresh rate on random-scan displays.

→ ~~The number of lines regulates refresh rate on random-scan displays.~~ An area of memory called refresh display files stores picture definition as a set of line drawing commands.

→ **Advantages :-**

- → Higher resolution as compared to raster Scan display.
- → Produces smooth line drawing.
- → Less Memory required.

→ **Disadvantages:**

- → Realistic images with different shades cannot be drawn.
- → colour limitations.

→ Differentiate between Random and Raster scan Display :-

| Random Scan | Raster Scan |
|---|---|
| 1. It has high resolution. | 1. Its resolution is low. |
| 2. It is more expensive. | 2. It is less expensive. |
| 3. Any modification if needed is easy | 3. Modification is tough. |
| 4. Solid Pattern is tough to fill. | 4. Solid pattern is easy to fill. |
| 5. Refresh rate depends on resolution. | 5. Refresh rate does not depend on the picture. |
| 6. Only screen with view on an area is displayed. | 6. Whole screen is scanned. |
| 7. Beam Penetration technology come under it. | 7. Shadow mark technology came under this. |
| 8. It is restricted to line drawing applications. | 8. It is suitable for realistic display. |

## → Colour CRT Monitors

→ A CRT monitor displays color pictures by using a combination of phosphors that emits different-colored light.

→ The emitted light from the different phosphors merges to form a single perceived color, which depends on the particular set of phosphors that have been excited.

→ There are two popular approaches for producing color displays with a CRT are:

— Beam Penetration Method
— Shadow-Mask Method

### 1. Beam Peneration Method:-

The beam-Penetration method has been used with random-scan monitors. In this method, the CRT screen is coated with two layers of phosphor, red and green and the displayed color depends on how far the electron beam penetrates the phosphor layers.

### 2. Shadow-Mask Method:-

→ Shadow Mask Method is commonly used in Raster-Scan System because they produce a much wider range of colours than the beam-penetration method.

→ It is used in the majority of color TV sets and monitors.

## Direct View Storage Tubes :-

DVST terminals also use ~~a to~~ the random scan approach to generate the image on the CRT screen. The term "storage tube" refers to the ability of the screen to retain the image which has been projected against it, thus avoiding the need to rewrite the image constantly.

→ Two guns are used in DVST :

1. Primary guns: It ~~is~~ stores the picture pattern.
2. Flood gun: maintains the picture display.

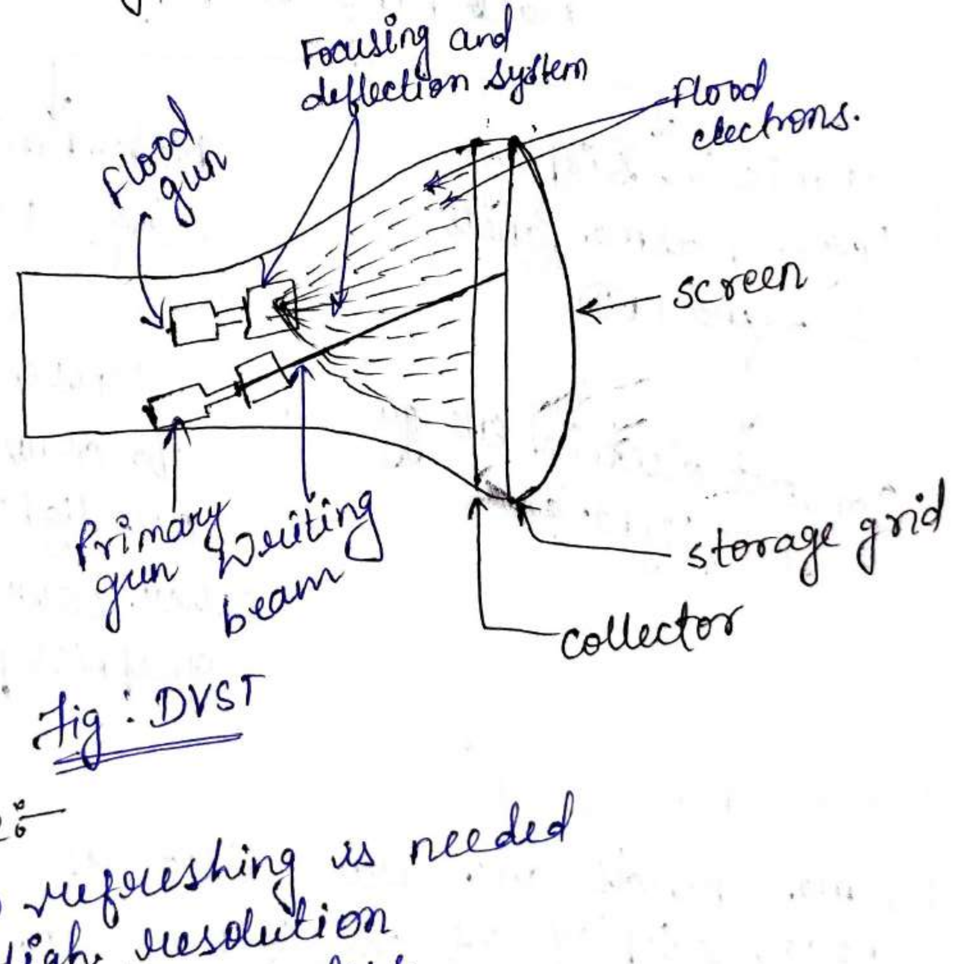



Fig : DVST

**Advantage:-**

1. No refreshing is needed
2. High resolution
3. Cost is very less.

**Disadvantage:-**

1. It is not possible to erase the selected part of a picture.

2. It is not suitable for dynamic graphics applications.
3. If a part ~~is~~ of picture is to modify, then time is consumed.

## ** Flat Panel Display :-

The Flat-Panel display refers to a class of video devices that have reduced volume, weight and power requirement compare to CRT.

e.g:- Small T.V. monitor, calculator, pocket video games,

Flate Panel Display

- Emissive Display (e.g:- Plasma Panels and LED)
  - Convert electrical energy into Light.
- Non-Emissive Display (e.g: LCD)
  - use optical effects to convert sunlight or light from some other source into graphics patterns.

## ** Plasma Panel Display :-

- Plasma - Panels are also called as Gas-Discharge Display. It consists of an array of small lights. Lights are fluorescent in nature.
- The essential components of the plasma-panel display are :-

(i) cathode (ii) Anode (iii) Fluorescent Cells
(iv) Glass plates

**LED (Light Emitting Diode):-**

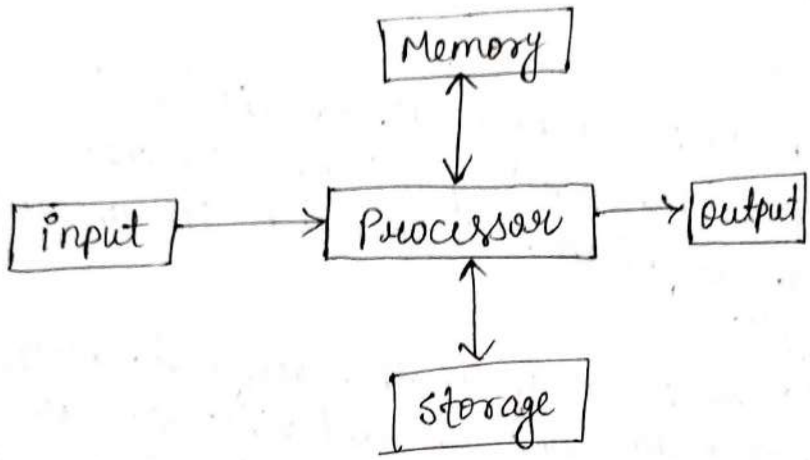
→ In an LED, a matrix of diodes is organized to form the pixel positions in the display and picture definition is stored in a refresh buffer.

→ Data is read from the refresh buffer and converted to voltage level that are applied to the diodes to produce the light pattern in the display.

**LCD (Liquid Crystal Display):-**

Liquid crystal Displays are the devices that produce a picture by passing polarized light from the surroundings or from an internal light source through a liquid-crystal material that transmits the light.

## ** Graphics Input/Output Devices:-




Q. A video monitor has a display area measuring 12 inch by 9.6 inch. If the resolution is 1280 X 1024 and the aspect ratio is 1. What is the ~~diagra~~ diameter of each screen point.

Sol^n:-
Length of monitor = L = 12 inches
Width " " = W = 9.6 "
Resolution of monitor = 1280 X 1024
pixels along length = 1280
pixels " width = 1024
Aspect ratio = r = 1

we know that
pixel on screen : length on monitor
when aspect ratio is 1 then
Considering the length
1280 pixels = 12 inches
$\Rightarrow$ 1 pixel = $12/1280$ = 0.009375 inches
Considering the width
~~12~~ 1024 pixels = ~~120~~ 9.6 inches

$\Rightarrow$ 1 pixel = $9.6/1024$ = 0.009375 inches.

So, diameter of point on screen = 0.009375 inches

**Q.2.** What is the fraction of the total refresh time per frame spent in retrace of the electron beam for a non-interlaced raster system with a resolution of 1280 X 1024, a refresh rate of 60 Hz, a horizontal retrace time of 5 $\mu$sec and a vertical retrace time of 500 $\mu$s?

**Sol$^n$:-** 1 sec = 10^6 $\mu$sec

Refresh rate = 60 Hz = $1/60$ sec to scan = 16.7 msec

The time for horizontal retrace = 1024 X 5 $\mu$sec

" " " Vertical " = 500 $\mu$sec

Total time spent for retrace = 5120 + 500 = 5620 $\mu$sec

= 5.62 msec.

The fraction of the total refresh time frame spent in retrace = $5.62/16.7$ = 0.337.

## → Input Devices:-

→ Keyboard
→ Mouse ⟶ enables the user to initiate actions.
→ Microphone
→ Digital Camera
→ Scanner

→ **Input devices:**

- Used by a person to communicate to a computer.

and **Output**

- textual info.
- Visual images - photos, diagrams, icons
- moving images
- Sounds - music, voice etc.

- O/P devices display info. from the computer to a person.
- The keyboard allows ~~the~~ comp. user to enter words, numbers, punctuation, symbols and special function commands into the computer's memory.
- Types of Mice
  - Wheel mouse
  - Cordless mouse.
    - uses infrared signals to connect to the computer's IrDA port.
- <u>Indirect Pointing Devices</u>:-
  - Need more cognitive processing than direct methods, but can be more efficient
  - mouse
  - tracker ball
  - track point
  - touchpad
  - Joystick
    - play comp games by controlling the way that something moves on the screen.
- <u>Microphones</u>:-
  - a speech recognition program that can process the i/p and convert it into machine-recognized commands or input.
- <u>Space ball</u> :-
  - pointing devices, moves in 6 dirⁿ.
  It used in CAD application & animations.

→ Digital camera:

↳ can store many more pictures than an ordinary camera. ~~Pictures taken using a digital ca~~ Takes pictures by converting the light passing through the lens at the front into a digital image.

→ Light pen:–

↳ used to choose objects or commands on the screen either by pressing it against the surface of the screen or by pressing a small switch on its side.

→ Bar codes:

→ It is a set of lines of different thicknesses that represent a number.

→ Bar code readers work by shining a beam of light on the lines.

→ O/P Devices:

– Visual o/p:– Text, graphics, and video.

– Audio o/p – sounds, music, and synthesized speech

Monitor, Printers, speakers, data projectors.

Types of monitors:

– cathode-ray-tube (CRT)

– Liquid Crystal Display (LCD).

Printers:

→ Ink Jet Printer
- Less expensive, colors, slower with a higher per page cost than laser printers.

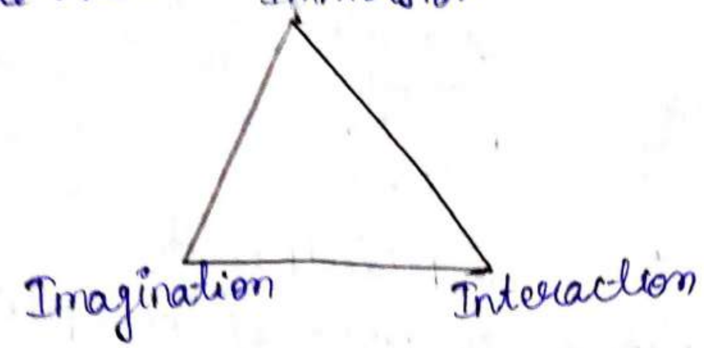
→ Laser printer
- More expensive, faster, lower per page cost than ink jet.

| Ink-jet printer | Laser printer |
|---|---|
| (1) also called bubble-jet, makes characters by inserting dots of ink onto paper. | (1) works like a copier. |
| (2) Letter-quality printouts. | (2) Quality determined by dots per inch produced. Color printer available. |
| (3) Cost of Printer is inexpensive but ink is costly. | (3) Expensive initial costs but cheaper to operate per page. |

→ Plotter:
↳ It is a printer that uses a pen that moves over a large revolving sheet of paper.

## → Virtual Reality i/o Devices

• VR Triangle or $I^3$

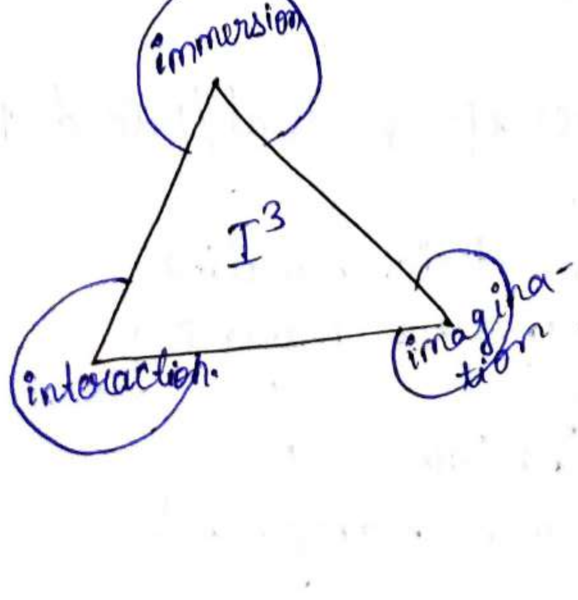



→ What is Virtual Reality?

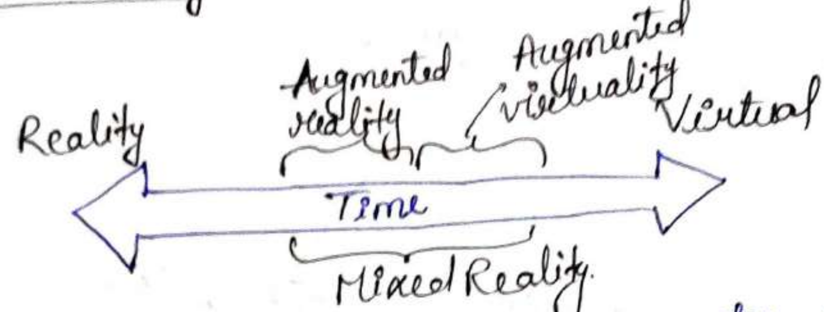
Definition : Virtual Reality is a high-end user-computer interface that involves real-time simulation and interactions through multiple sensorial channels. These sensorial modalities are ~~test~~ visual, auditory, tactile, smell and taste.

"It is a computer-simulated reality which replicates an environment, real or imagined and simulates a user's physical presence and environment to allow for user interaction".

Virtual Reality triangle:-



** Mixed Reality Timeline




Mixed reality is an area of computer research that deals with the combination of real-world and computer-generated data, where computer-generated objects are visually mixed into the real environment and vice versa in real time.

→ Augmented Reality :-

→ It is a virtuality environment, means virtual content in real environment.

→ AR is a live direct or indirect view of a physical, real-world environment whose elements are augmented or ~~sp~~ supplemented by computer-generated sensory input such as sound, video, graphics or GPS data.

→ Techniques of Virtual/Mixed Reality:

· Modeling:
- Geometric modeling
- Kinematics modeling
- Physical "
- Behaviour "
- Model management

- Selection and Manipulation
- Travel
  - Wayfinding.
- Hand Dataglove
- Head Mounted Display
- Eye Tracker

→ Virtual Reality tools and packages:

- Cortona 3D
- VRML
- OpenSceneGraph
- WorldToolkit
- Java 3D
- APIs
- AR Toolkits

→ Traditional application
Traditional applications.
- Medical applications
- Education, Arts, and Entertainment
- Military VR applications

→ Emerging applications of VR:
- VR applications in manufacturing
- Applications of VR in Robotics
- Information Visualization.

## ** Raster-Scan Systems:-

→ Interactive raster-graphics systems typically employ several processing units.

→ In addition to the CPU, a special-purpose processor, called the video controller or display controller, is used to control the operation of the display device.

→ Here, the frame buffer can be anywhere in the system memory, and video controller accesses the frame buffer to refresh the screen.

### Video Controller:-

→ A fixed area of the system memory is reserved for the frame buffer and the video controller is given direct access to the frame-buffer memory.

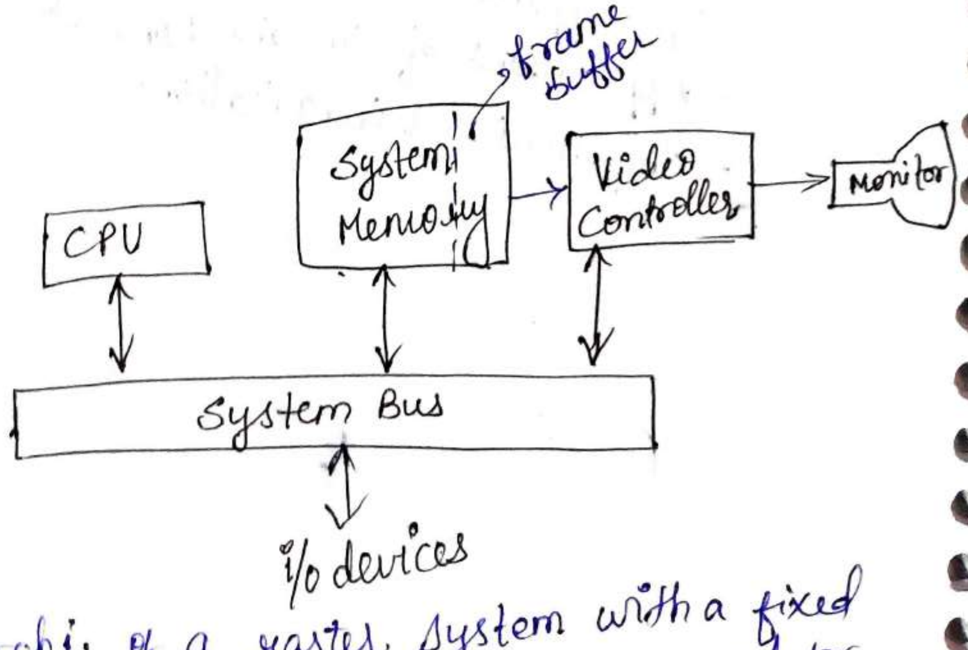



fig. Archi. of a raster system with a fixed portion of the system memory reserved for the frame buffer.

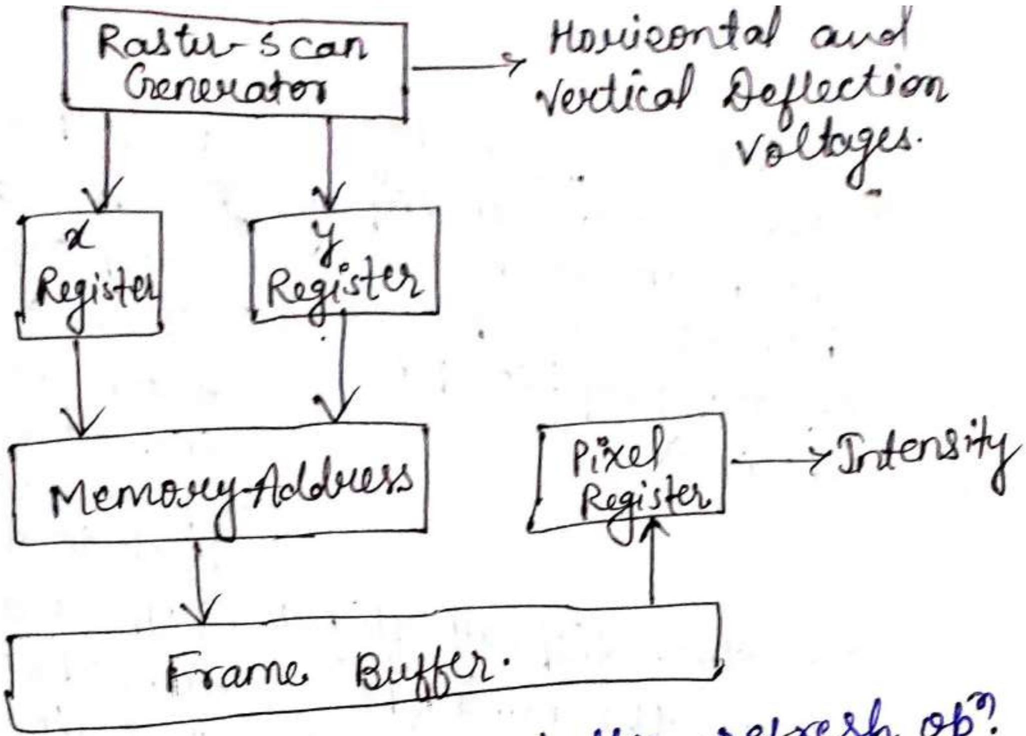



fig :- Basic video-controller refresh op?

## ** Raster-Scan Display Processor :-

→ A major task of the display processor is digitizing a picture definition given in an application program into a set of pixel values for storage in the frame buffer.

→ This digitization process is called scan conversion also known as rasterization.

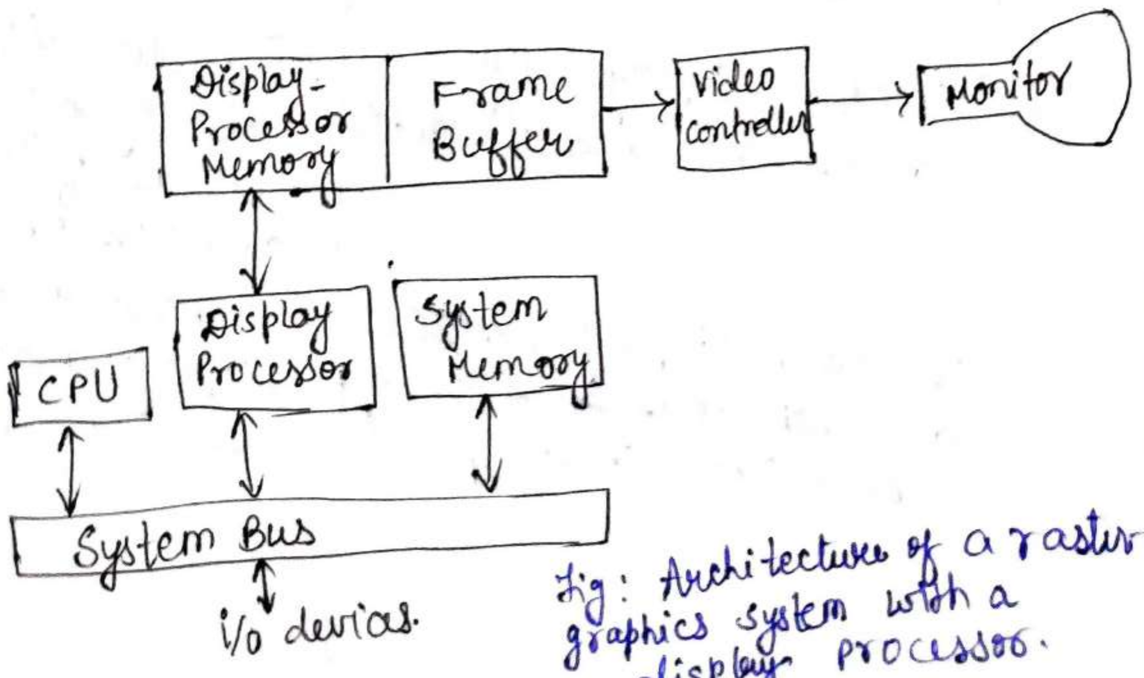
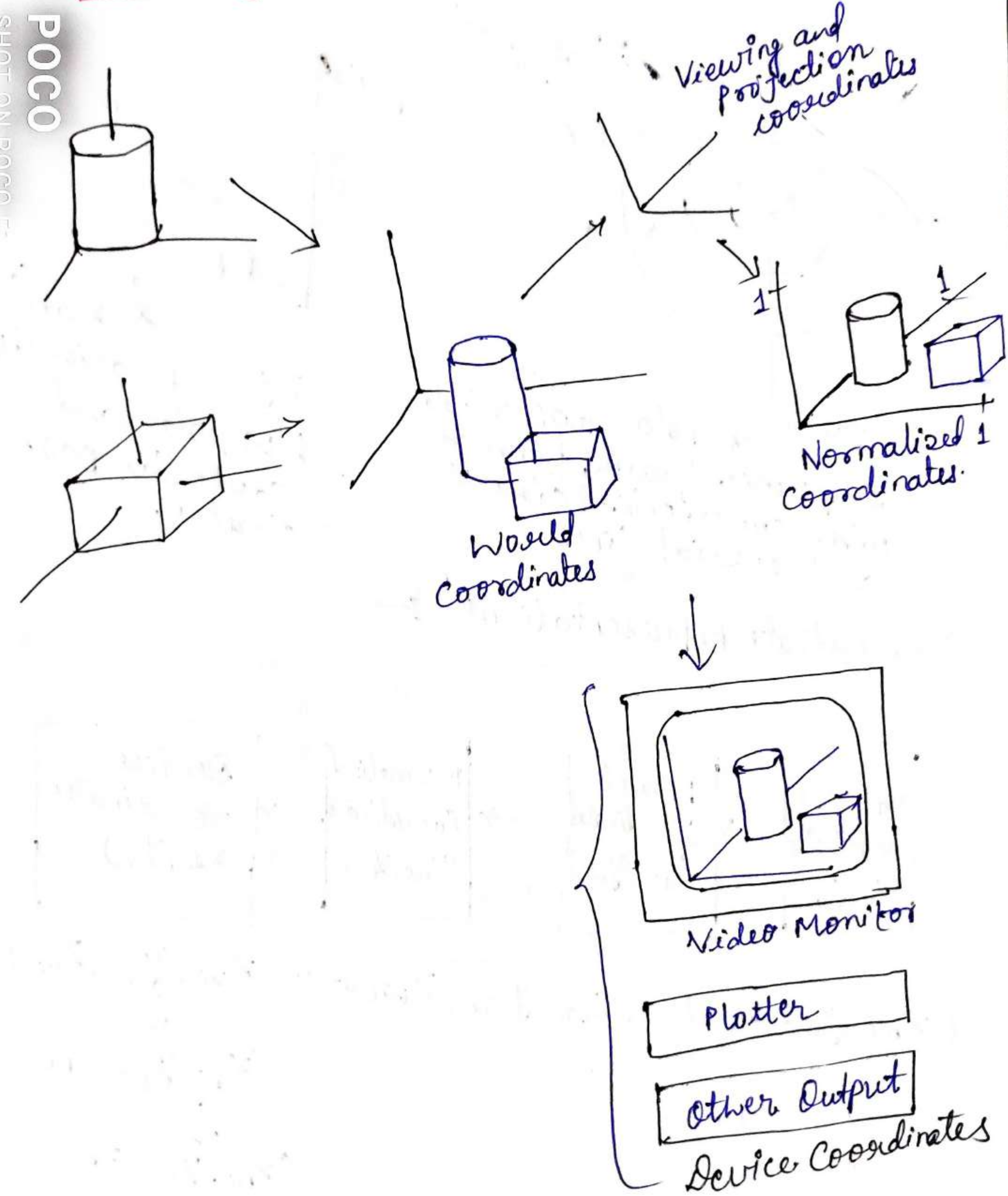



fig : Architecture of a raster graphics system with a display processor.

## ** Vector Scan Vs Raster Scan Display :-

| Vector Scan Display | Raster Scan Display. |
| --- | --- |
| (i) In vector scan display the beam is moved b/w the end points of the graphics primitives. | (1) In raster scan display the beam is moved all over the screen one scan line at a time, from top to bottom and then back to top. |
| (2.) Vector display flickers when the no. of primitives in the buffer becomes too large. | (2) In raster display, the refresh process is independent of the complexity of the image. |
| (3.) Scan conversion is not required. | (3.) Graphics primitives are specified in terms of their endpoints and must be scan converted into their corresponding pixels in the frame buffer. |
| (4.) Cost is more. | (4.) Cost is low. |
| (5) Vector display only draws lines and characters | (5.) Raster display has ability to display areas filled with solid colours or patterns. |
| (6) Vector display draws a continuous and smooth lines. | (6.) Raster display can display mathematically smooth lines, polygons. |



Viewing and projection coordinates
1
1
1
Normalized coordinates.
World Coordinates
Video Monitor
Plotter
Other Output
Device Coordinates

## * Polar Coordinate Reference Frame :-

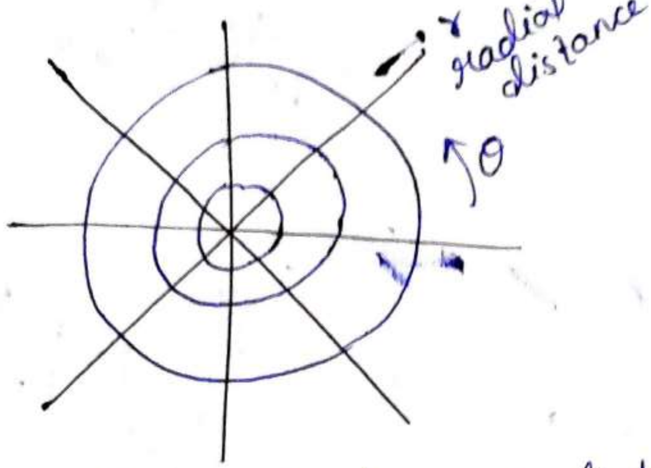



Fig: 2 A polar-coordinate reference frame, formed with concentric circles and radial lines.

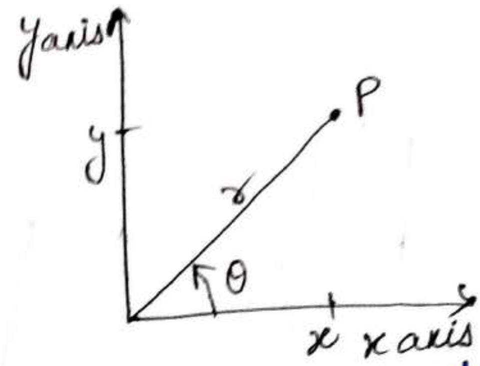



Fig 3: Relationship b/w polar and Cartesian coordinates.

### → Co-ordinate Representations :-

Modeling Coordinates $(x_{mc}, y_{mc})$ → World Coordinates $(x_{wc}, y_{wc})$ → Normalized Coordinates $(x_{nc}, y_{nc})$ → Devices coordinates $(x_{dc}, y_{dc})$

$(x_{mc}, y_{mc}, z_{mc}) \rightarrow (x_{wc}, y_{wc}, z_{wc}) \rightarrow (x_{vc}, y_{vc}, z_{vc})$

$\downarrow$

$(x_{pc}, y_{pc}, z_{pc})$

$\downarrow$

$(x_{nc}, y_{nc}, z_{nc})$

$\downarrow$

$(x_{dc}, y_{dc})$.

## ++ Interactive and Passive Graphics:-

→ Interactive Computer Graphics:

→ In interative Computer graphics user have some controls over the picture, i.e. the user can make any change in the produced image.

e.g. ping-pong game.

→ Interative Computer graphics require two-way communication b/w the computer and the user. A User can see the imge and make any change by sending his command with an i/p device.

→ Non-Interactive Or Passive Computer-graphics :-

→ In non-interactive computer graphics, the picture is produced on the monitor, and the user does not have any controlled over the image, i.e. the user cannot make any change in the rendered image.

e.g.:- Titles shown on T.V.

→ It involves only one-way communication b/w the computer and the user, User can see the produced image, and he cannot make any change in the image.

## ** Graphics Software and Libraries:-

→ Software Standards:

- The primary goal of standardized graphics s/w is portability.
- When packages are designed with standard graphics functions, software can be moved easily from one hardware system to another and used in different implementations and applications.
- Without standards, programs designed for one h/w system often cannot be transfered to another system without extensive rewriting of the programs.

Graphics packages:

- GKS in 1984
- PHIGS and PHIGS+
- SGI
- OpenGL architecture review board
- Language binding
- Open Inventor
- Virtual Reality Modeling Language (VRML).
- Java2D
- Java3D
- RenderMan Interface
- Mathematica
- Matlab
- Maple

## ** Primitives : Basic Graphics Primitives:



I. Scan Conversion a line.
II. Scan Conversion Circle.
III. Scan conversion Ellipse.

→ Filled area primitives.

### → Scan Conversion

Scan Conversion is the process of representing continuous graphics objects as a collection of discrete pixels. Each pixel can have either on or off state.

The video display device of the computer represent the picture by converting binary values (0,1) into a pixel on and pixel off information. 0 is represented by pixel off. 1 is represented using pixel on.

→ Scan Conversion:- It is a technique for changing the vertical / horizontal scan frequency of video signal for different purposes and applications.

→ The device which performs this conversion is called Scan converter. Scan conversion is also known as Rasterization.

→ The application of scan conversion is video projectors, cinema equipment, standard and HDTV televisions, LCD monitors, radar display etc.

→ There are two method for scan conversion:

1) Analog Methods:
In this method, the conversion is done using large numbers of delay cells and is appropriate for analog video.

2) Digital Method:
In this method, the picture is stored in frame buffer with $x1$ speed and is read with $x2$ speed.

→ Advantage of developing algorithms for scan conversion:–

1. Algorithms can generate graphics objects at a faster rate.
2. Using algorithms memory can be used efficiently.
3. Algorithms can develop a higher level of graphical objects.

→ examples of objects which can be scan converted.

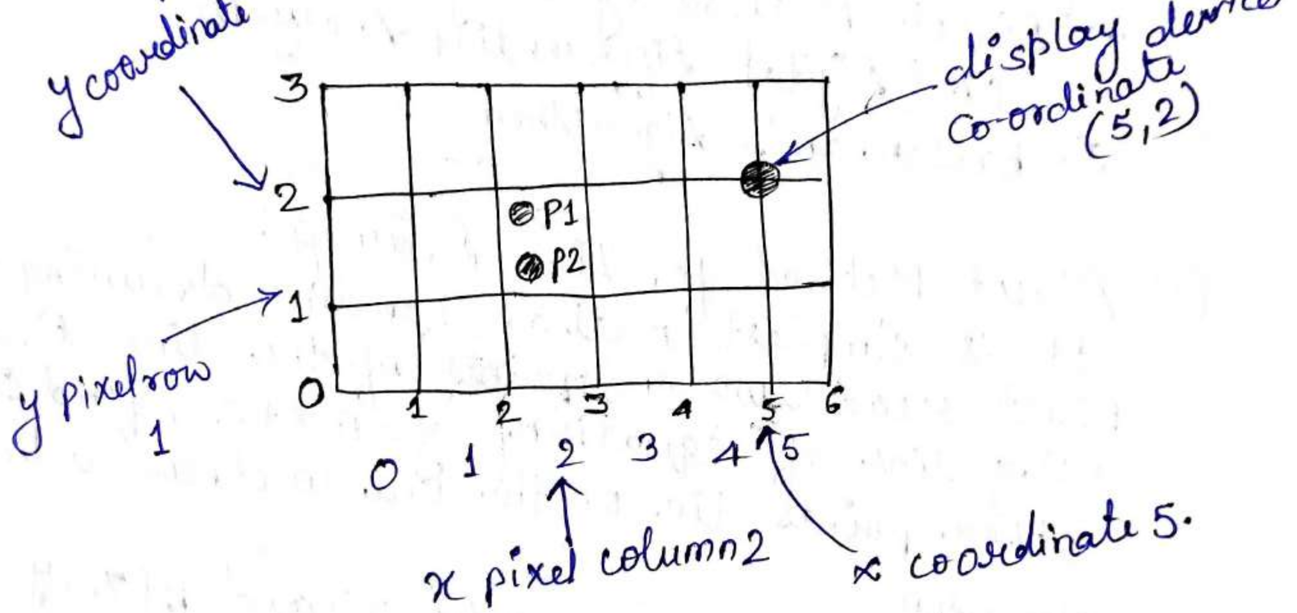
1. Point
2. Line
3. sector
4. Arc
5. Ellipse
6. Rectangle
7. Polygon
8. Filled Regions.

## ** Scan Converting a Point

→ Each pixel on the graphics display does not represent a mathematical point.

→ Instead, it means a region which theoretically can contain an infinite number of points.

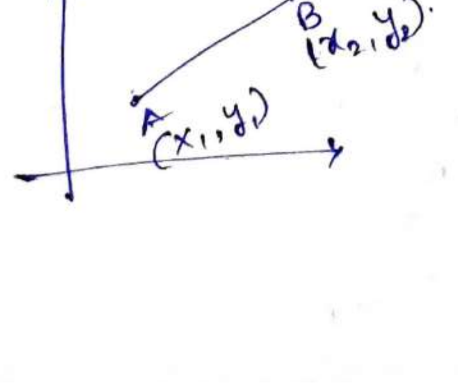
→ Scan-converting a point involves illuminating the pixel that contains the point.




## ** Scan Converting a line :-

What is Line?

- A line can be defined as a straight one-dimensional figure. It has no thickness and can be extends endlessly in both directions. A line segment is only a part of a line. It is described as the distance b/w any two points.



Equation of Line

$$y = mx + c$$

$$m = \Delta y / \Delta x$$

$$\Delta x = x2 - x1 \quad , \quad \Delta y = y2 - y1$$

In computer Graphics, Line can be drawn by plotting the pixels in sequence.

→ Algorithm for Line Drawing :—

1) Direct Method by using Line Equation
2) DDA (Digital Differential Analyzer)
3) Bresenham's Algorithm

(1) Direct Method for Line Drawing:

It is simplest method for line drawing. First read two end points of the line then using line of equation $y = mx + c$, find out other points lie on the line to draw a line.

Algorithm

Step 1: Read two end points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$

Step 2: Calculate

$$dx = x_2 - x_1$$

$$dy = y_2 - y_1$$

Step 3: Now $m = \frac{dy}{dx}$

Step 4: Set $(x, y)$ to starting point

if $dx > 0$ then

$x = x_1$

$y = y_1$

$x_{end} = x_2$

if $dx < 0$ then

$x = x_2$

$y = y_2$

$x_{end} = x_1$

$c = y - mx$

Step 6: Plot a point at current $(x, y)$ coordinates.

Step 7: Increment value of $x$

$$x = x + 1.$$

Step 8: Compute value of $y$

$$y = mx + c$$

Step 9: If $x = x_{end}$ then stop otherwise go to step 6

e.g. Draw a line using direct method with end points $(0,0)$ and $(8,24)$.

Sol: Step 1: We have two end points $(0,0)$ and $(8,24)$

i.e. $x_1 = 0$, $y_1 = 0$ $x_2 = 8$ $y_2 = 24$

Step 2: $dx = x_2 - x_1 = 8$

$dy = y_2 - y_1 = 24$

Step 3: $m = dy/dx = 24/8 = 3$.

Step 4: Set initial point $(x, y)$

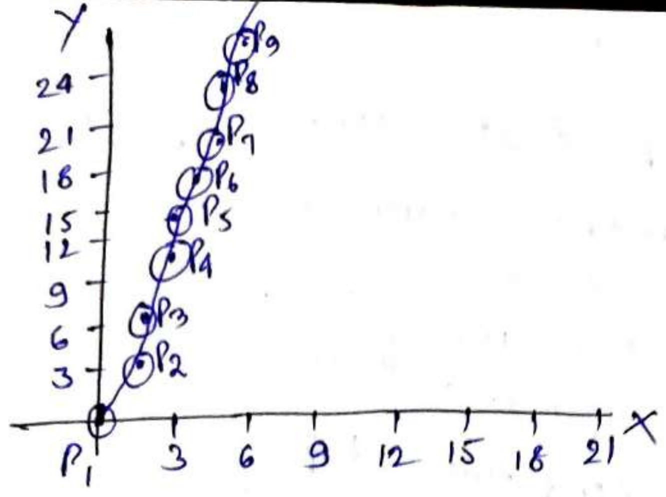
We have $dx > 0$

so, $x = 0$, $y = 0$, $x_{end} = 8$

Step 5: Calculate $c = y - mx$ ; $c = 0$.

Step 6, 7, and 8 execute until $x = x_{end}$.

| $x = x+1$ | $y = mx + c$ | Plotting Points |
|---|---|---|
| 0 | 0 | $P_1(0,0)$ |
| 1 | 3 | $P_2(1,3)$ |
| 2 | 6 | $P_3(2,6)$ |
| 3 | 9 | $P_4(3,9)$ |
| 4 | 12 | $P_5(4,12)$ |
| 5 | 15 | $P_6(5,15)$ |
| 6 | 18 | $P_7(6,18)$ |
| 7 | 21 | $P_8(7,21)$ |
| 8 | 24 | $P_9(8,24)$ |




## ** DDA Algorithm :-

What is DDA?

Digital Differential Analyzer (DDA) algorithm is an incremental scan conversion method of line drawing. DDA is used for linear interpolation of variables over an interval between start and end point. DDA is used for rasterization of lines, triangles and polygons.

→ Algorithm

Step 1: Read two end points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$.

Step 2: Find approximate length of the line

if $(ABS(x_2 - x_1) > ABS(y_2 - y_1))$ then

$length = ABS(x_2 - x_1)$

otherwise

$Length = ABS(y_2 - y_1)$.

Step 3: Find raster unit

$dx = (x_2 - x_1)/Length$

$dy = (y_2 - y_1)/length$

Step 4: Set $x = x_1$, $y = y_1$ and $i = 0$

Step 5: Now plot the point $(x, y)$

$x = x + dx$

$y = y + dy$

Step 6: Repeat step 5 until $i <= \text{length}$

Step 7: stop

**Advantage:** — (1) This method gives overflow indication when a point is repositioned.

1) It is simplest algorithm and easy to implement.
2) DDA is faster method than Direct Method of line drawing.
3) DDA does not use multiplication theorem.

**Disadvantage:** — (2) It is more suitable for generating line using the s/w. But it is less suited for h/w implementation.

1) Floating point arithmetic in DDA algo is still time ~~cons~~ consuming.

e.g. Draw a line with end points (0,0) and (8,8) using DDA algorithm.

Step 1: We have two end points (0,0) and (8,8)

i.e. $x_1 = 0 \quad y_1 = 0 \quad x_2 = 8, \; y_2 = 8$

Step 2: $abs(x_2 - x_1) = 8$

$abs(y_2 - y_1) = 8$

So length $= 8$.

Step 3: $dx = 1, \; dy = 1$

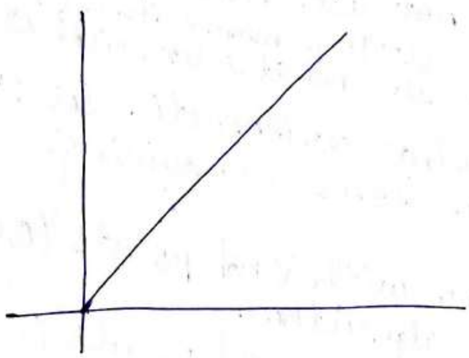
Step 4: Set Initial point $(x, y)$

$x = 0$

$y = 0$

Step 5: execute until $i <= \text{length}$.

| i | x | y | Plotting points |
|---|---|---|---|
| 0 | 0 | 0 | $P_1(0,0)$ |
| 1 | 1 | 1 | $P_2(1,1)$ |
| 2 | 2 | 2 | $P_3(2,2)$ |
| 3 | 3 | 3 | $P_4(3,3)$ |
| 4 | 4 | 4 | $P_5(4,4)$ |
| 5 | 5 | 5 | $P_6(5,5)$ |
| 6 | 6 | 6 | $P_7(6,6)$ |
| 7 | 7 | 7 | $P_8(7,7)$ |
| 8 | 8 | 8 | $P_9(8,8)$ |



## ** Bresenham's Line drawing Algorithm:

This algorithm is accurate and efficient Line generating algorithm. ~~It was developed by Jack Elton~~. It scan converts line using only incremental integer calculations.

**Algorithm :-**

Step 1: Read two end points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$.

Step 2: Calculate

$$dx = x_2 - x_1$$
$$dy = y_2 - y_1$$

Step 3: Calculate decision parameter P

$$P = 2dy - dx$$

Step 4: Set initial point

$x = x_1,\ y = y_1$ and $i = 0$

Step 5: Now plot the point $(x, y)$

if $P < 0$ then

$x = x + 1$

$P = P + 2dy$

otherwise,

$x = x + 1$

$y = y + 1$

$P = P + 2dy - 2dx$

Step 6: Repeat step 5 until $i <= dx$.

Advantage :-

1.) It is simple because it involves only integer arithmetic.

2.) It avoids the generation of duplicates points.

3.) It is faster than DDA because it does not involve floating point calculations.

(4) It can be implemented using h/w bcoz it does not use multiplication & division

Disadvantage :-

(1) This algorithm is for basic line drawing.

(2) It always assume that $x_1 < x_2$

(3) We can not draw vertical line using Bresenham's line drawing algorithm.

e.g. Draw a line with end points (20,10) and (30,18) using Bresenham's algorithm.

Step 1: We have two end points (20,10) and (30,18)

i.e. $x_1 = 20$ $y_1 = 10$, $x_2 = 30$, $y_2 = 18$.

Step 2: $dx = x_2 - x_1 = 10$

$dy = y_2 - y_1 = 8$

Step 3: Calculate decision parameter

$P = 2dy - dx = 6$

Step 4: set initial point $(x, y)$

$x = x_1 = 20$

$y = y_1 = 10$

and $i = 0$

Step 5: execute until $i <= dx$

| i | P | Plotting Points |
|---|---|---|
| 0 | | (20,10) |
| 1 | 6 | (21,11) |
| 2 | 2 | (22,12) |
| 3 | -2 | (23,12) |
| 4 | 14 | (24,13) |
| 5 | 10 | (25,14) |
| 6 | 6 | (26,15) |
| 7 | 2 | (27,16) |
| 8 | -2 | (28,16) |
| 9 | 14 | (29,17) |
| 10 | 10 | (30,18) |

## ** Difference b/w DDA and Bresenham's Line Algorithms:

| DDA Algorithm | Bresenham's Line algo. |
|---|---|
| 1. DDA algo. use floating point, i.e. Real Arithmetic. | 1. Bresenham's Line algo use fixed point, i.e. Integer Arithmetic. |
| 2. DDA algo uses multiplication & division its operation | 2. It uses only subtraction and addition its operation. |
| 3. It is slower than Bresenham's Line Algo in line drawing bcoz it uses real arithmetic. | 3. It is faster than DDA algo in line bcoz it involves only addition & subtraction in its calculation and uses only integer arithmetic. |
| 4. DDA algorithm is not accurate and efficient as Bresenham's line algorithm. | 4. ~~Bre~~ It is more accurate and efficient at DDA algo. |
| 5. It can draw circle and curves but are not accurate as Bresenham's line algo. | 5. It can draw circle and curves with more accurate ~~as~~ than DDA algo. |

*** Antialiasing of lines:—**

→ The aliasing effect is the appearance of jagged edges or "jaggies" in a rasterized image (an image rendered using pixels):

→ The problem of jagged edges technically occurs due to distortion of the image when scan conversion is done with sampling at a low frequency, which is also known as undersampling.

→ Aliasing occurs when real-world objects which comprise of smooth, continuous curves are rasterized using pixels.

→ Antialiasing is a technique used in computer graphics to remove the aliasing effect.

→ The aliasing effect can be reduced by adjusting intensities of the pixels along the line.

**Methods of Antialiasing:—**

→ ① Increasing resolution.

→ ② Unweighted area sampling

↳ The intensity of pixel is proportional to the amount of line area occupied by the pixel. This technique produces noticeably better results than does setting pixels either to full intensity or to zero intensity.

③ Weighted area sampling:

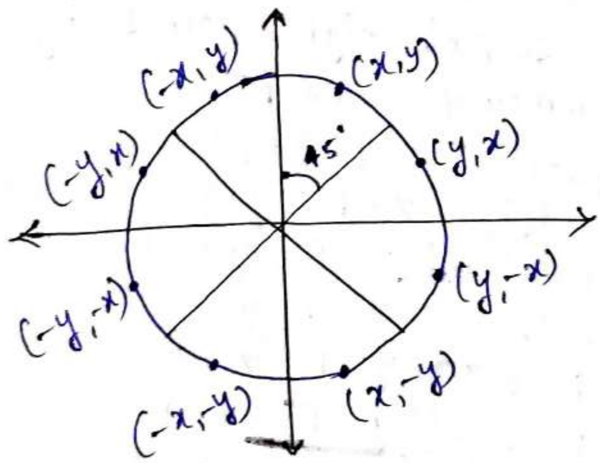
In weighted area sampling equal areas contribute unequally i.e. a small areas closer to the pixel center has greater intensity than does one at a greater distance. Thus, the intensity of the pixel ~~ed~~ is dependent on the line area occupied and the distance of area from the pixel's center.

## ** Scan Conversion a Circle:-

- A circle is defined as the set of points that are all at a given distance $r$ from a center position $(x_c, y_c)$.
- The circle is divided into 4 quadrant, each quadrant has two octant. so circle is also known as eight-way symmetric figure.

→ For drawing, we consider that circle centered at origin (0,0). If a point $(x,y)$ lies in one octant then other seven points can be calculated easily by using eight-way symmetry.

$(x,-y)$ $(-x,-y)$ $(-x,y)$ $(y,x)$ $(y,-x)$ $(-y,-x)$ $(-y,x)$




Most popular circle drawing algorithms :-

1) Bresenham's algorithm
2) Midpoint Circle Algorithm

## (1.) Bresenham's Circle Drawing algorithm:

This algorithm selects nearest pixel position to complete arc. It is very significant algorithm and faster than other circle drawing algorithm because it uses only integer arithmetic.

Algorithm:

Step 1: I/P center of the circle $(p, q)$

Step 2: I/P radius of the circle $r$

Step 3: Calculate

$d = 3 - 2r$ → decision variable

Step 4: Initialize

$x = 0$

$y = r$

Step 5: Now, center is at $(p, q)$ and current pixel is $(x, y)$ so plot eight points by using eight-way symmetry concept:

putpixel $(x+p, y+q)$
putpixel $(x+p, -y+q)$
putpixel $(-x+p, -y+q)$
putpixel $(-x+p, y+q)$
putpixel $(y+p, x+q)$
putpixel $(y+p, -x+q)$
putpixel $(-y+p, -x+q)$
" $(-y+p, x+q)$

Step 6: Find next pixel location

if $d <= 0$ then

$x = x + 1$

$d = d + 4x + 6$

Otherwise

$x = x + 1$

$y = y - 1$

$d = d + 4(x - y) + 10$

step 7: if $x <= y$ then go to step 5
otherwise
stop algo.

**Advantage:**

① It is simple algorithm.
② It can be implement easily.

**Disadvantage:**

① There is the problem of accuracy while generating points.
② This algorithm is not suitable for the complex and high graphic images.

## → DDA Circle Drawing algorithm:

→ The equation of circle, with origin as the Center of the circle is given as: $x^2+y^2=r^2$

→ The DDA algorithm can be used to draw the circle by defining circle as a differential eqⁿ.

$2x\,dx + 2y\,dy = 0$ where $r$ is constant

$\therefore x\,dx + y\,dy = 0$

$\therefore y\,dy = -x\,dx$

$\therefore \frac{dy}{dx} = -\frac{x}{y}$

**Algo:**

1. Read the radius ($r$), of the circle and calculate value of $\epsilon$
2. start_x = 0
   start_y = r
3. $x_1$ = start_x
   $y_1$ = start_y

4. do{

$x_2 = x_1 + \epsilon y_1$

$y_2 = y_1 - \epsilon x_2$

plot (int($x_2$), int($y_2$))

$x_1 = x_2$;

$y_1 = y_2$;

[Reinitialize the current point]

} while($y_1$ - start_y) < $\epsilon$ or (start_x - $x_1$) > $\epsilon$

[Check if the current point is the starting point or not. If current point is not starting point repeat step 4; otherwise stop].

5. stop.

** Midpoint circle drawing algorithm :-

The Midpoint circle drawing algorithm use to determine the points needed for rasterising a circle.

The Midpoint algorithm calculates all the points in the first octant of the circle and then prints them along with their mirror points in the other seven octants. [bcoz circle is a symmetric fig].

→ Mid-point circle algorithm based on the equation of circle $x^2+y^2=r^2$

$$P = f(x,y) = x^2+y^2-r^2$$

First point is $(x,y) = (0,r)$

Next point of first octant will be the midpoint of $(x+1, y)$ and $(x+1, y-1)$.

i.e. $(x+1, y-0.5)$

POCO
SHOT ON POCO F1

put this point into equation of circle:

$$P = (x+1)^2 + (y-0.5)^2 - r^2$$

To define decision parameter put $(0, r)$.

$$P = 1 - r$$

if $P < 0$ then

next point $(x+1, y)$

$$P = P + 2x + 3$$

if $P >= 0$ then

next point $(x+1, y-1)$

$$P = P + 2(x-y) + 5$$

## Algorithm

Step1: Input center of the circle $(xc, yc)$.

Step2: Input radius of the circle $r$.

Step3: Calculate decision parameter

$$P = 1 - r$$

Step4: Initialize first point

$$x = 0, \; y = r.$$

Step5: Now, center is at $(xc, yc)$ and current pixel is $(x, y)$ so plot eight point by using eight-way symmetry concept.

putpixel $(x+xc, y+yc)$
putpixel $(x+xc, -y+yc)$
putpixel $(-x+xc, -y+yc)$
putpixel $(-x+xc, y+yc)$
putpixel $(y+xc, x+yc)$
putpixel $(y+xc, -x+yc)$
putpixel $(-y+xc, -x+yc)$
putpixel $(-y+xc, x+yc)$.

Step 6: Find next pixel
if $P<0$ then
$x = x+1$
$P = P+2x+3$
Otherwise
$x = x+1$
$y = y-1$
$P = P+2(x-y)+5$

Step 7: if $x <= y$ then
go to step 5
otherwise
stop.

Advantages:-
1) It is very effective and efficient algorithm.
2) The algorithm is based on the equation of circle $x^2+y^2=r^2$
3) It can be implement easily.

Disadvantages:-
1) There is the problem of accuracy while generating points.
2) This algorithm is time consuming.

## ** Scan conversion of Ellipse:—

→ An ellipse is an elongated circle.

→ Therefore, elliptical curves can be generated by modifying circle-drawing procedures to make into account the different dimensions of an ellipse along the major and minor axes.

→ The ellipse is also a symmetric figure like a circle but is four-way symmetry rather than eight-way.

→ Two methods of defining an Ellipse:
- Polynomial method
- Trigonometric method

→ **Polynomial method for Ellipse:**

• The ellipse has a major and minor axis. If $a_1$ and $b_1$ are major and minor axis respectively. The center of ellipse is $(i, j)$. The value of $x$ will be incremented from $i$ to $a_1$ and value of $y$ will be calculated using the following formula

$$y = b_1\sqrt{1 - \frac{x-i}{a_1^2}} + j$$

→ Drawback of Polynomial method:

① It requires squaring of values. So floating point calculation is required.

② Routines developed for such calculations are very complex and slow.

→ Trigonometric method for Ellipse :-

→ The following equation defines an ellipse trigonometrically as shown in fig:

$$x = a * \cos(\theta) + h \text{ and}$$

$$y = b * \sin(\theta) + k$$

Where $(x, y)$ = the current coordinates
$a$ = length of major axis
$b$ = " " minor "
$\theta$ = current angle
$(h, k)$ = ellipse center

→ In this method, the value of $\theta$ is varied from 0 to $\pi/2$ radians. The remaining points are found by symmetry.

→ Drawback of Trigonometric method:
1. This is an inefficient method.
2. It is not an interactive method for generating ellipse.
3. The table is required to see the trigonometric value.
4. Memory is required to store the value of $\theta$.

Topics remain
→ Midpoint ellipse drawing algo.
→ scan-conversion of conic section.
→ scan-converting a Polygon

POCO
SHOT ON POCO

## ** Seed Fill

→ The seed fill algorithm is further classified as flood fill and boundary fill algo.

Seed Fill

### Flood fill

1. An algo that fill interior-defined regions.
2. In this method, a point or seed which is inside region is selected. This point is called a seed point.
3. Then four connected approaches or eight connected approaches is used to fill with specified color.
4. We start from a specified interior point (x, y) and reassign all pixel values are currently set to a given interior color with the desired color.
5. Using either a 4-connected or 8-connected approaches, we then step through pixel positions until all interior points have been repainted.

### boundary fill (edge-fill)

1. An algo that fill boundary-defined regions.
2. It uses the recursive method. A starting pixel called as the seed is considered.
3. This algo checks boundary pixel or adjacent pixels are colored or not.
4. If the adjacent pixel is already filled or colored then leave it, otherwise fill it.
5. The filling is done using 4 connected or eight connected approaches.
6. Four connected approaches is more suitable than the eight connected approach.

Disadvantage
① Very slow algo.
② May be fail for large polygons
③ Initial pixel required more knowledge about surrounding pixels.

Disadvo
① This algo takes time and memory bcoz some recursive calls are needed.

→ Boundary fill algo.

```
Void boundaryFill(int x, int y, int fill,
                  int boundary) {
 int current;
 current = getPixel(x,y);
 if ((current != boundary) && (current != fill)) {
     setColor(fill);
     setPixel(x, y);
   boundaryFill(x+1, y, fill, boundary);
       "      (x-1, y,  "  ,    "    );
       "      (x , y+1, "  ,    "    );
       "      (x , y-1, "  ,    "    );
 }
}
```

POCO
SHOT ON POCO F1

Flood fill algo

```
void floodFill(int x, int y, int fillColor, int
                                              oldColor){

    if (getPixel(x,y)== oldColor){
        setColor( fillColor);
        setPixel(x,y);

        floodfill(x+1, y, fillColor, oldColor);
           "      (x-1, y,     "     ,    "    );
           "      (x , y+1,    "     ,    "    );
           "      (x , y-1,    "     ,
    }
}.
```

→ Scan Line Polygon Fill Algorithm:

# 2D Transformation

$\Downarrow$

changing of an object after creation.
{in terms of position or size}.

## Different Types of Transformation

- **Translation** → changing the position of the object
- **Rotation** → Rotating the object with an angle $\theta$.
- **Scaling** → increasing/decreasing the size of the object
- **Reflection** → Mirroring the object w.r.t given object
- **Shearing** → changing/tilting/skewing the shape of the object.

## Translation:-

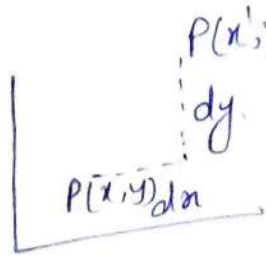
→ It will shift the object from one position to other position.

→ To translate a pt from coordinate position $(x, y)$ to another $(x_1, y_1)$ we add algebrically the translation distances $T_x$ & $T_y$ to original coordinates.

$T_x$ & $T_y$ is called shift vector.

$$x_1 = x + T_x$$
$$y_1 = y + T_y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix}$$



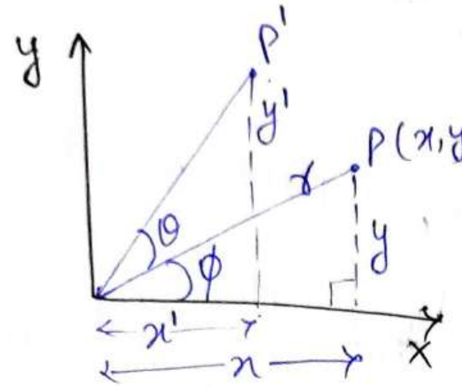

$$\boxed{P' = P + T}$$

$T \rightarrow$ Translating factor.

→ **Rotation** :—

→ It is a process of changing the angle of the object.

→ Rotation can be clockwise or anticlockwise.




ϕ is the angle b/w x-axis & point P.

r = distance from origin to P.

Let coordinate of P in the polar form.

$x = r\cos\phi$

$y = r\sin\phi$

then $x' = r\cos(\phi+\theta)$

$y' = r\sin(\phi+\theta)$

$x' = r[\cos\phi\cos\theta - \sin\phi\sin\theta]$

$= r\cos\phi\cos\theta - r\sin\phi\sin\theta$

$$\boxed{x' = x\cos\theta - y\sin\theta}$$

[put $r\cos\phi = x$ & $r\sin\phi = y$].

$y' = r[\sin\phi\cos\theta + \cos\phi\sin\theta]$

$= r\sin\phi\cos\theta + r\cos\phi\sin\theta$

$$\boxed{y' = y\cos\theta + x\sin\theta}$$

⇒ so $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$

↳ rotation matrix in anticlockwise ation

⇒ $\boxed{P' = R \cdot P}$

POCO
SHOT ON POCO F1

for clockwise dirⁿ,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

e.g. One triangle is given (2,2) (8,2), (5,5)
Rotate the triangle by 90°.

solⁿ:-

$$R = \begin{bmatrix} \cos 90^\circ & -\sin 90^\circ \\ \sin 90^\circ & \cos 90^\circ \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

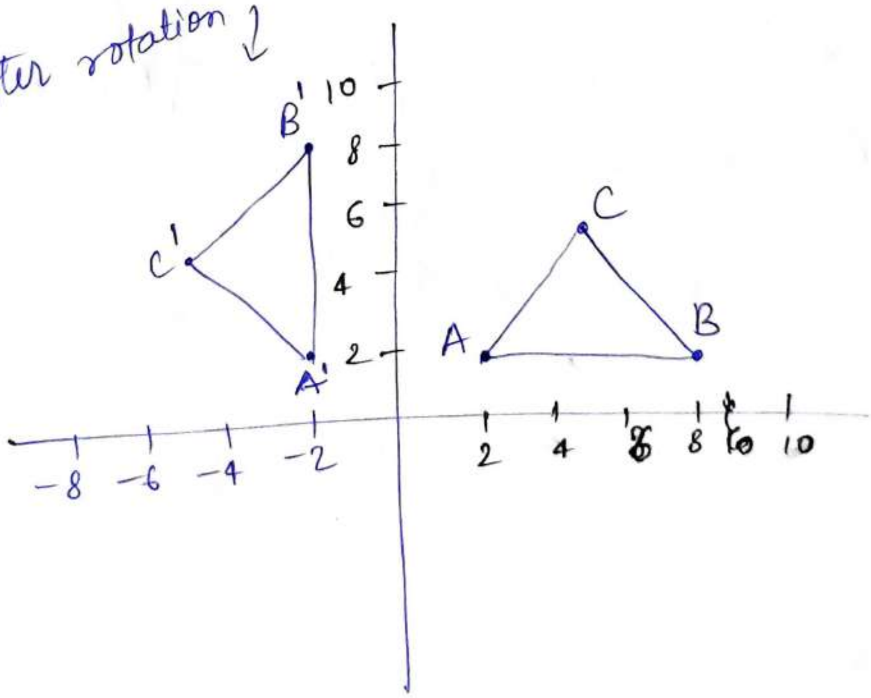
$$P' = R \cdot P$$

$$A' = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \end{bmatrix}$$

$$B' = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 8 \\ 2 \end{bmatrix} = \begin{bmatrix} -2 \\ 8 \end{bmatrix}$$

$$C' = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 5 \\ 5 \end{bmatrix} = \begin{bmatrix} -5 \\ 5 \end{bmatrix}$$

after rotation



## → Scaling :-

↳ It is used to change the size of object.

There are two scaling factors.

$S_x \rightarrow x$ dir$^n$

$S_y \rightarrow y$ dir$^n$.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\boxed{P' = S \cdot P}$$

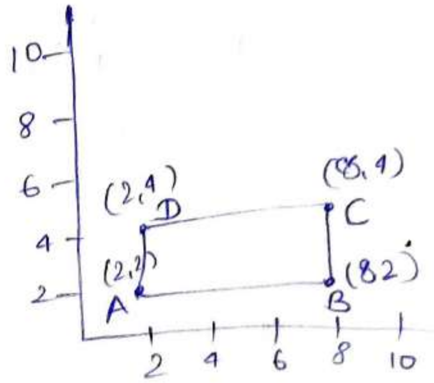
if $S_x, S_y = 1$, object size will not change

if $S_x, S_y > 1$, object size enlarge

if $S_x, S_y < 1$, object size reduce.

e.g. Make the size of the given object double.

$sf = 2$.




$p' = S \cdot P$ , $S = 2$.

$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

$S_x = 2$, $S_y = 2$

$A' = \begin{bmatrix} 4 \\ 4 \end{bmatrix}$, $B' = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \cdot \begin{bmatrix} 8 \\ 2 \end{bmatrix}$

$= \begin{bmatrix} 16 \\ 4 \end{bmatrix}$

$C' = \begin{bmatrix} 16 \\ 8 \end{bmatrix}$, $D' = \begin{bmatrix} 4 \\ 8 \end{bmatrix}$.

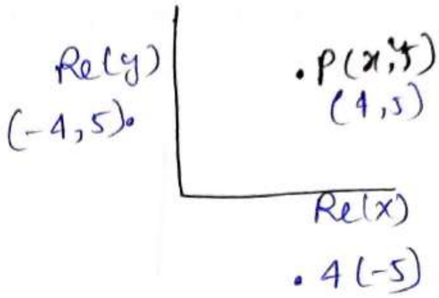
## * Reflection:

It is a transformation which evaluate the mirror image of the object.

→ It is of two types.

(i) Reflection w.r.t x-axis

(ii) Reflection w.r.t y-axis




Reflection by x-axis

⇒ Re(x)

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

($P'$ = Re(x) · P)

$$\boxed{P' = Re(x) \cdot P}$$

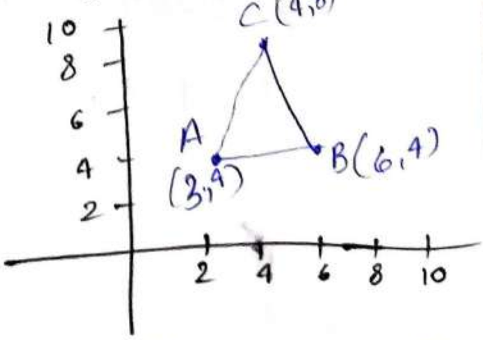
⇒ Re(y)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \boxed{P' = Re(y) \cdot P}$$

e.g. A triangle ABC is given, the coordinates A, B, C are given as

A(3,4) B(6,4) C(4,8)

find reflected position of triangle i.e. to the x-axis.




Sol^n The matrix for Re(x) is

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

→ The 'A' pt co-ordinates after reflection.

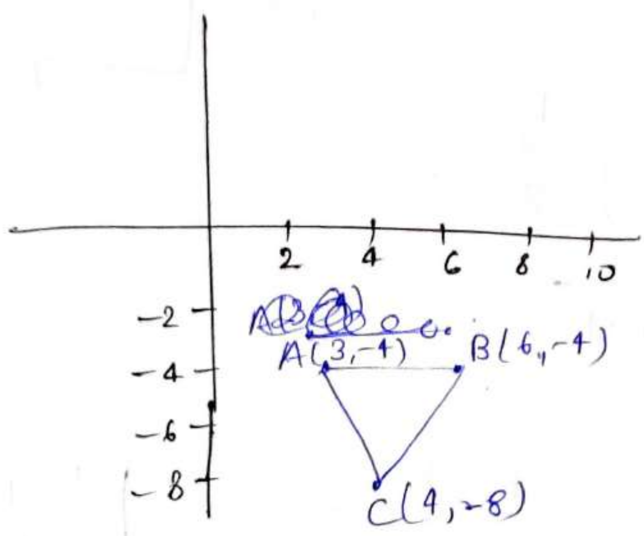
$$(x, y) = [3, 4] \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [3, -4]$$

→ The 'B' pt co-ordinates after reflection

$$(x, y) = [6, 4]\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [6, -4]$$

→ The 'C' pt co-ordinates after reflection.

$$(x, y) = [4, 8]\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = [4, -8]$$




→ 2D- shearing.

It is similar to sliding the layers in one dir$^n$ to change shape of object.

x-shear :- Here y- coordinates remains same

If $P(x, y)$ new Points $P(x', y')$

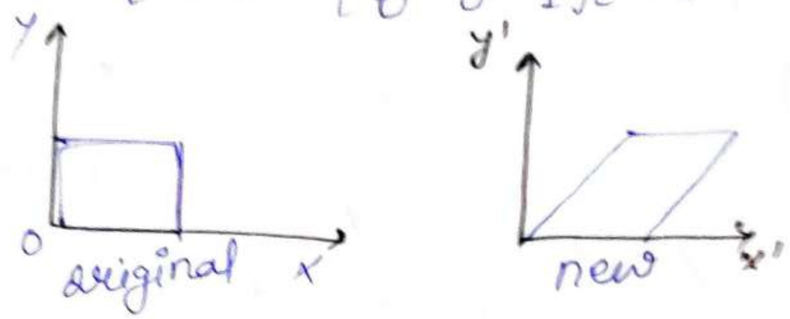
$$x' = x + sh_x * y$$
$$y' = y$$

Matrix form,

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix}$$

homogeneous Coordinates

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ Shx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$




## y-shear

↳ shearing along y-axis.

x-remains same.

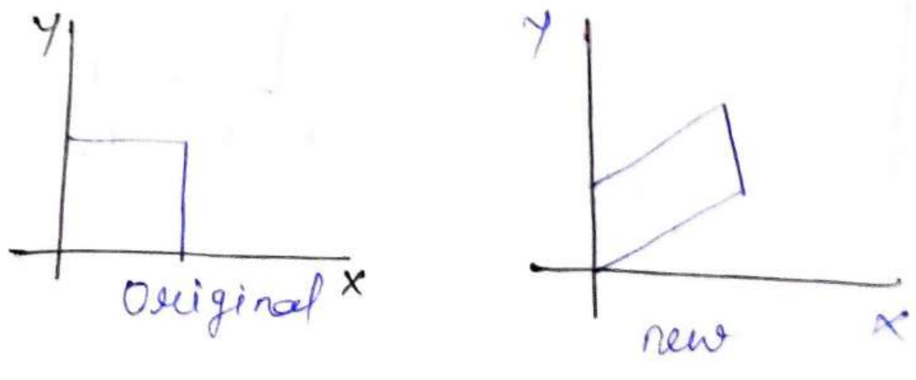
$P(x,y) \Rightarrow P'(x', y')$

$$x' = x$$
$$y' = y + shy \cdot x$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ shy & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Homogeneous Coordinate:-

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & shy & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$



e.g. A triangle with (2,2), (0,0) & (2,0). Apply shearing factor 2 on x-axis and 2 on y-axis. Find out the new coordinates of triangle.

Sol$^n$: $A(2,2)$, $B(0,0)$, $C(2,0)$, $Sh_x = 2$, $Sh_y = 2$

x-shear :

$A = (x_1, y_1)$ $\quad A(x_0, y_0) = (2,2)$

$x_1 = x_0 + Sh_x \cdot y_0 = 2 + 2*2 = 6$

$y_1 = y_0 = 2.$ $\quad \boxed{(6,2)}$

$B = (0,0)$ $\quad B = (x_1, y_1)$
$\;\;\;\; x_0 \; y_0$

$x_1 = x_0 + Sh_x \cdot y_0 = 0 + 2 \cdot 0 = 0$

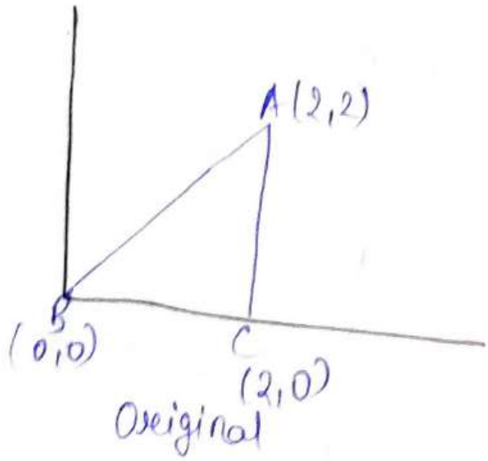
$y_1 = y_0 = 0$ $\quad \boxed{(0,0)}$

$C = (x_0, y_0) = (2,0)$ $\quad C = (x_1, y_1)$

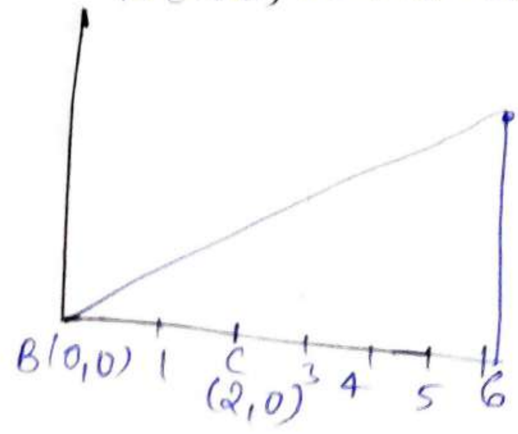
$x_1 = x_0 + Sh_x \cdot y_0$

$= 2 + 2 \cdot 0 = 2$

$y_1 = y_0 = 0$ $\quad \boxed{(2,0)}$




A(6,2), B(0,0), C(2,0)



y-shear in coordinate A (2,2)

$x_1 = x_0 = 2$

$y_1 = y_0 + sh_y \cdot x_0 = 2 + 2 * 2 = 6.$

for co ord. B(0,0)

$x_1 = x_0 = 0$

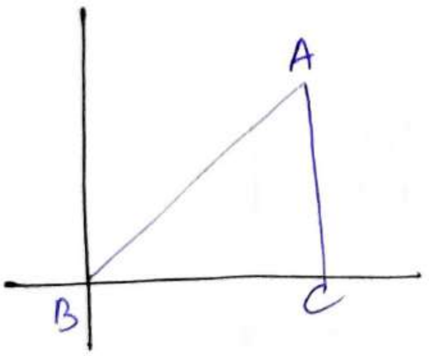
$y_1 = y_0 + sh_y \cdot x_0 = 0 + 2 \cdot 0 = 0.$

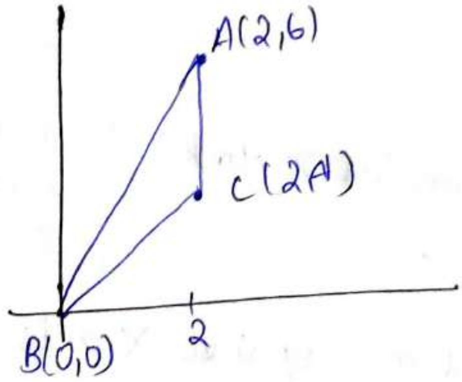
for coord. , C(2,0)

$x_1 = x_0 = 2$

$y_1 = y_0 + sh_y \cdot x_0 = 0 + 2 * 2 = 4$

$\boxed{(2,4)}$



→ Following are matrix for two-dimensional transformation in homogeneous co-ordinate:

1. Translation $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$ or $\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$

2. Scaling $\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$

3. Rotation (Anti Clockwise) $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

4. Rotation (~~anti~~ clock) $\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

5. Reflection against X axis $\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

6. " " Y axis $\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

7. " " Origin $\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

8. " " line $Y = X$ $\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

9. " " " $Y = -X$ $\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

10. Shearing in X dir$^n$ $\begin{bmatrix} 1 & 0 & 0 \\ Sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

11. Shearing in Y dir$^n$ $\begin{bmatrix} 1 & Sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

12. " " both X & Y dir$^n$ $\begin{bmatrix} 1 & Sh_y & 0 \\ Sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$.

## → Two-dimensional Composite Transformations

We can set up a sequence of transformations as a Composite transformation matrix by calculating the product of the individual transformations.

## → General Two-Dimensional Pivot-Point Rotation

When a graphics package provides only a rotate function with respect to the coordinate origin, We can generate a two-dimensional rotation about any other pivot point $(x_r, y_r)$ by performing the following sequence of translate-rotate-translate operations:

1. Translate the object so that the pivot-point position is moved to the coordinate origin.
2. Rotate the object about the co-ordinate origin.
3. Translate the object so that the pivot point is returned to its original position.

→ The composite transformation matrix for this sequence is obtained with the concatenation.

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

## → General Two-Dimensional Fixed-Point Scaling:

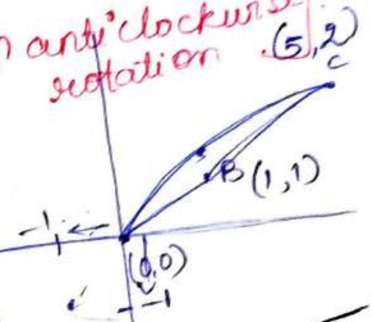
1. Translate the object so that the fixed point coincides with the co-ordinate origin
2. Scale the object with respect to the coordinate origin.
3. Use the inverse of the translation in Step (1) to return the object to its original position.

$$\begin{bmatrix}1 & 0 & x_f\\0 & 1 & y_f\\0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}S_x & 0 & 0\\0 & S_y & 0\\0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1 & 0 & -x_f\\0 & 1 & -y_f\\0 & 0 & 1\end{bmatrix}$$

$$=\begin{bmatrix}S_x & 0 & x_f(1-S_x)\\0 & S_y & y_f(1-S_y)\\0 & 0 & 1\end{bmatrix}$$

**Q.** Perform 45° rotation of a $\Delta$ $A(0,0)$, $B(1,1)$ $C(5,2)$ about the point $(-1,-1)$.

Sol$^n$: [if not mentioned then perform anti'clockwise rotation]



Translation Matrix

$$=\begin{bmatrix}1 & 0 & t_x\\0 & 1 & t_y\\0 & 0 & 1\end{bmatrix}=\begin{bmatrix}1 & 0 & -1\\0 & 1 & -1\\0 & 0 & 1\end{bmatrix}$$

Coeff. matrix of triangle

$$=\begin{bmatrix}0 & 0 & 1\\1 & 1 & 1\\5 & 2 & 1\end{bmatrix}$$

$$\text{Rotation matrix}=\begin{bmatrix}\cos 45^\circ & -\sin 45^\circ & 0\\\sin 45^\circ & \cos 45^\circ & 0\\0 & 0 & 1\end{bmatrix}$$

$$\begin{bmatrix} & & \\ & & \\ & & \end{bmatrix}\begin{bmatrix}1 & 0 & 1\\0 & 1 & 1\\0 & 0 & 1\end{bmatrix}P=\begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & 0\\1/\sqrt{2} & 1/\sqrt{2} & 0\\0 & 0 & 1\end{bmatrix}$$

$$\text{Transformation} = T\cdot R\cdot T'$$

$$=\begin{bmatrix}1 & 0 & -1\\0 & 1 & -1\\0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & 0\\1/\sqrt{2} & 1/\sqrt{2} & 0\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 1\\0 & 1 & 1\\0 & 0 & 1\end{bmatrix}$$

$$=\begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & -1\\1/\sqrt{2} & 1/\sqrt{2} & -1\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}1 & 0 & 1\\0 & 1 & 1\\0 & 0 & 1\end{bmatrix}=\begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & -1\\1/\sqrt{2} & 1/\sqrt{2} & \sqrt{2}-1\\0 & 0 & 1\end{bmatrix}$$

$$\begin{bmatrix}1/\sqrt{2} & -1/\sqrt{2} & 0\\1/\sqrt{2} & 1/\sqrt{2} & 0\\1-\sqrt{2} & 1 & 1\end{bmatrix}\cdot\begin{bmatrix}0 & 0 & 1\\1 & 1 & 1\\5 & 2 & 1\end{bmatrix}$$

$$\begin{bmatrix}-1/\sqrt{2} & -1/\sqrt{2} & \sqrt{2}\\1/\sqrt{2} & 1/\sqrt{2} & 3-\sqrt{2}\\6 & 3 & \end{bmatrix}$$

$1-\sqrt{2}\ 1\ 1$

$0$

→ Two-dimensional Composite Transformations

We can set up a sequence of transformations as a Composite transformation matrix by calculating the product of the individual transformations.

→ General Two-Dimensional Pivot Point Rotation

When a graphics package provides only a rotate function with respect to the coordinate origin, we can generate a two-dimensional rotation about any other pivot point $(x_r, y_r)$ by performing the following sequence of translate rotate translate operations:

1. Translate the object so that the pivot point position is moved to the coordinate origin.
2. Rotate the object about the co-ordinate origin.
3. Translate the object so that the pivot point is returned to its original position.

→ The composite transformation matrix for this sequence is obtained with the concatenation

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

→ General Two-Dimensional Fixed-Point Scaling:

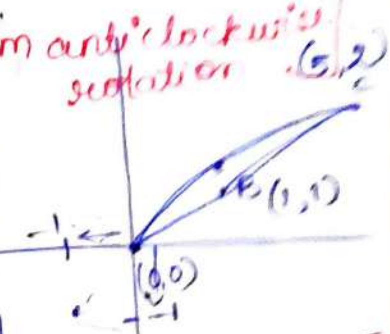
1. Translate the object so that the fixed point coincides with the co-ordinate origin.
2. Scale the object with respect to the coordinate origin.
3. Use the inverse of the translation in step (1) to return the object to its original position.

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} S_x & 0 & x_f(1-S_x) \\ 0 & S_y & y_f(1-S_y) \\ 0 & 0 & 1 \end{bmatrix}$$

Q. Perform $45^\circ$ rotation of a $\Delta$ $A(0,0)$, $B(1,1)$, $C(5,2)$ about the point $(-1,-1)$.

Sol$^n$: [If not mentioned then perform anticlockwise rotation




Translation Matrix

$$= \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rotation matrix} = \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 0 \\ \sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Coeff. matrix of triangle

$$= \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 5 & 2 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} P = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\text{Transformation} = T \cdot R \cdot T'$$

$$= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & \sqrt{2}-1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1-\sqrt{2} & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 5 & 2 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & -1/\sqrt{2} & \sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} & 3-\sqrt{2} \\ 1/\sqrt{2} & 3 & \\ 6 & & \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 5 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & \sqrt{2}-1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ \sqrt{2} & 0 & \sqrt{2}-1 \\ 7/\sqrt{2} & \frac{-3}{\sqrt{2}} & -6+2\sqrt{2} \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & \sqrt{2}-1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 4+\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} & 1+\sqrt{2} \\ \sqrt{2} & & \end{bmatrix}$$

$$\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & \sqrt{2}-1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -1 & \frac{3}{\sqrt{2}}-1 \\ \sqrt{2}-1 & 2\sqrt{2}-1 & \frac{9}{\sqrt{2}}-1 \\ 1 & 1 & 1 \end{bmatrix}$$

$\frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}}$

$\frac{5}{\sqrt{2}} - \frac{2}{\sqrt{2}} - 1$

$\frac{3}{\sqrt{2}}$ $\quad \frac{2}{\sqrt{2}} - \frac{7}{\sqrt{2}} + \sqrt{2} - 1$

$A' = (-1, \sqrt{2}-1)$, $B' = (-1, 2\sqrt{2}-1)$

$C' = \left(\frac{3}{\sqrt{2}} - 1, \frac{9}{\sqrt{2}} - 1\right)$.

**Q.** Perform a 45° rotation of a triangle $A(0,0)$, $B(1,1)$, $C(5,2)$ about an arbitrary point $P(-1,-1)$.

[by default anticlockwise].

**Sol^n^:**

$$[A \;\; B \;\; C] = \begin{array}{ccc} A & B & C \end{array}$$
$$\begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix}$$

rotation about $(-1,-1)$.

$$= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45^\circ & -\sin 45^\circ & 0 \\ \sin 45^\circ & \cos 45^\circ & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & +1 \\ 0 & 1 & +1 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}}_{\text{first multiply}}$$

$$= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & 0 \\ 1/\sqrt{2} & 1/\sqrt{2} & 2/\sqrt{2} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & 2/\sqrt{2}-1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Matrix after rotation —

$$= \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} & -1 \\ 1/\sqrt{2} & 1/\sqrt{2} & \frac{2}{\sqrt{2}}-1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 5 \\ 0 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -1 & \frac{3}{\sqrt{2}}-1 \\ \frac{2}{\sqrt{2}}-1 & \frac{4}{\sqrt{2}}-1 & \frac{9}{\sqrt{2}}-1 \\ 1 & 1 & 1 \end{bmatrix}$$

So, $A' = (-1, \sqrt{2}-1)$, $B' = (-1, 2\sqrt{2}-1)$

$C' = \left( \frac{3\sqrt{2}}{2}-1, \frac{9}{2}\sqrt{2}-1 \right)$.

Q. Consider a square $A(1,0)$, $B(0,0)$, $C(0,1)$, $D(1,1)$. Rotate Pt by $45°$ clockwise about the point $A(1,0)$.

Sol$^n$:-

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos 45° & \sin 45° & 0 \\ -\sin 45° & \cos 45° & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

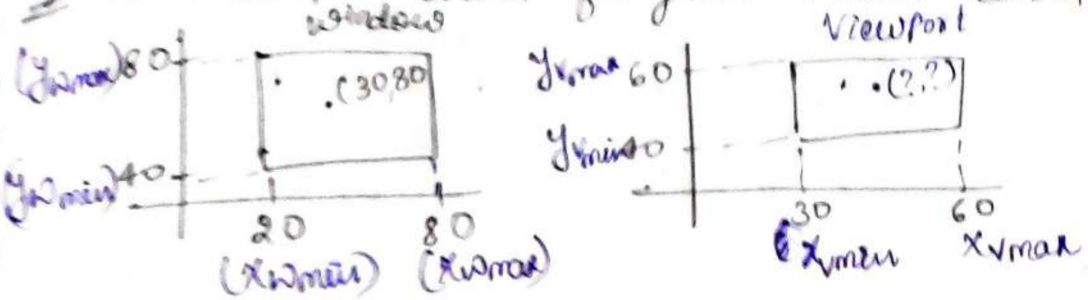
$$= \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1-1/\sqrt{2} & -1/\sqrt{2} & 1 \end{bmatrix}$$

Matrix after rotation:—

$$\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 1-1/\sqrt{2} & -1/\sqrt{2} & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1/\sqrt{2} & 0 & 1/\sqrt{2} & \sqrt{2} \\ & & & \\ & & & \end{bmatrix}$$

Q. Find viewport coord. for given window coord.




Given, $x_w = 30$, $y_w = 80$

$\longrightarrow x_v = ?$, $y_v = ?$

$x_{wmin} = 20$; $x_{wmax} = 80$ | $x_{vmin} = 30$, $y_{vmin} = 40$

$y_{wmin} = 40$, $y_{wmax} = 80$ | $x_{vmax} = 60$, $y_{vmax} = 60$

$$S_x = \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}} = \frac{60-30}{80-20} = \frac{30}{60} = \frac{1}{2}$$

$$S_y = \frac{Y_{vmax} - Y_{vmin}}{Y_{wmax} - Y_{wmin}} = \frac{60-40}{80-40} = \frac{20}{40} = \frac{1}{2}$$

Now,

$$x_v = X_{vmin} + (X_w - X_{wmin}) S_x$$
$$= 30 + (30-20) \times \frac{1}{2} = 35$$

$$y_v = Y_{vmin} + (Y_w - Y_{wmin}) S_y$$
$$= 40 + (80-40) \times \frac{1}{2} = 60$$

So, the point on window (30,80) will be (35,60) on viewport.

POCO
SHOT ON POCO F1

→ 2 D viewing

**Two dimensional viewing (The viewing pipeline):**

A world coordinate area selected for display is called a window. An area on a display device to which a window is mapped is called a view port. The window defines what is to be viewed, the view port defines where it is to be displayed. The mapping of a part of a world coordinate scene to device co-ordinate is referred to as viewing transformation. The two dimensional viewing transformation is referred to as window to view port transformation of windowing transformation.

**The two dimensional viewing transformation pipeline:-**

→ MC → Construct World-Coord. Scene Using Modeling-Coordinate transformations → WC → Convert World-Coordinates to Viewing Co-ordinates → VC → Map Viewing Coordinates to Normalised Viewing coordinates using Window-Viewport Specifications → NVC → Map Normalized Viewport to device coordinates → DC

The viewing transformation in several steps, as indicated in Fig. First, we construct the scene in world coordinates using the output primitives. Next to obtain a particular orientation for the window, we can set up a two-dimensional viewing-coordinate system in the world coordinate plane, and define a window in the viewing-coordinate system.

→ Once the viewing reference frame is established, we can transform descriptions in world coordinates to viewing coordinates. We then define a viewport in normalized coordinates (in the range from 0 to 1) and map the viewing-coordinate description of the scene to normalized coordinates.

→ At the final step all parts of the picture that lie outside the viewport are clipped, and the contents of the viewport are transferred to device coordinates. By changing the position of the viewport, we can view objects at different positions on the display area of an o/p device.

## 2D Clipping

The procedure that identifies the portion of a picture that are either inside or outside of a specified region of space is referred to as clipping. The region against which an object is to be clipped is called a clip window or clipping window.

→ The clipping ~~wind~~ algorithm determines which points, lines or portions of lines lie within the clipping window. These points, lines or portions of lines are retained for display.

1. Point clipping
2. Line clipping
3. Area clipping
4. Curve clipping
5. Text clipping

### 1) Point clipping :

The points are said to be interior to the clipping if $xw_{min} <= x <= xw_{max}$

$yw_{min} <= y <= yw_{max}$

The equal sign indicates that points on the window boundary are included within the window.

~~** Sutherland - Hodgeman polygon clip~~

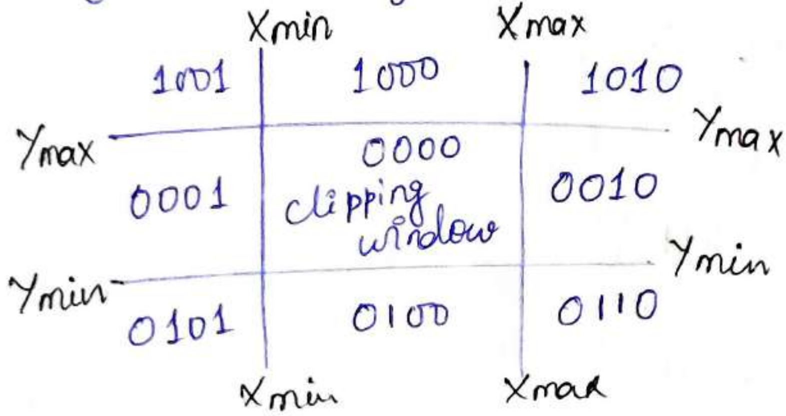
### ** Sutherland and cohen subdivision line clipping algorithm:-

It is fast line clipping algorithm. Processing time is reduced in the cohen-Sutherland method by performing more tests before processing proceeding to the intersection calculations.

→ every line endpoint in a picture is assigned a four-digit binary value, called a region code, and each bit position is used to indicate whether the point is inside or outside one of the clipping-window boundaries.

→ A value of 1 (or true) in any bit position indicates that the endpoint is outside that window border.

→ Similarly, a value of 0 (or false) in any bit position indicates that the endpoint is not outside (it is inside or on) the corresponding window edge.



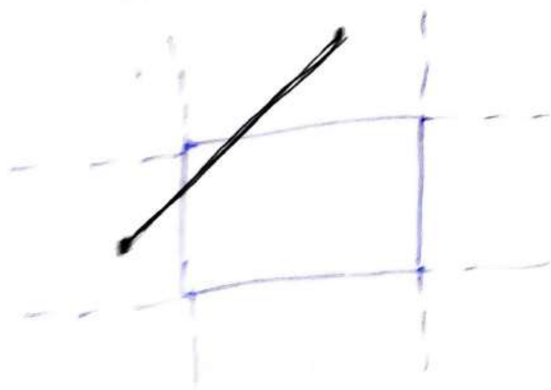

(i) generate TBRL (Region code) code → 4 bit code

Top bottom left Right

if $x < x_{min}$ → left of window
$x > x_{max}$ → right " "
$y < y_{min}$ → bottom " "
$y > y_{max}$ → top " "

Pseudo Code:

1. Assign the region code for 2 end points of a given line.
2. If both have region codes 0000 then line accepted completely.
3. else
   perform logical AND operation for both region codes.
   (a) If result ≠ 0000 line is outside
   (b) else line is partially inside
      (i) Choose an endpoint of the line that is outside the given rectangle.
      (ii) find intersection point
      (iii) Replace endpoint with the intersection point & update region code.
      (iv) Repeat step 2 until line is trivially accepted or trivially rejected.
4. Repeat step 1 for other lines.



Algo

(i) assign Region code to both end points say $c_0$ and $c_1$.

(ii) if $c_0$ OR $c_1$ == 0000 then accepted completely.
(inside window)

(iii) else if

$c_0$ AND $c_1$ $\neq$ 0000 Reject it

else

clip if line crossed $x_{min}$ or $x_{max}$

then

$y = y_1 + m(x - x_1)$

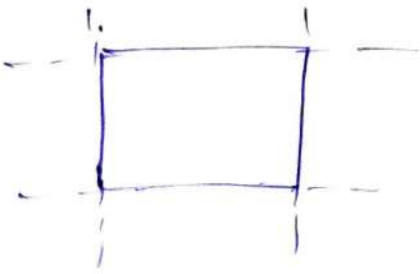
else

$x = x_1 + \frac{1}{m}(y - y_1)$

$x = x_{max}$ or $x_{min}$

$y = y_{max}$ or $y_{min}$

(iii) verify $x_{min} <= x \leq x_{max}$, $y_{min} \leq y \leq y_{max}$ } if it doesn't satisfy then repeat



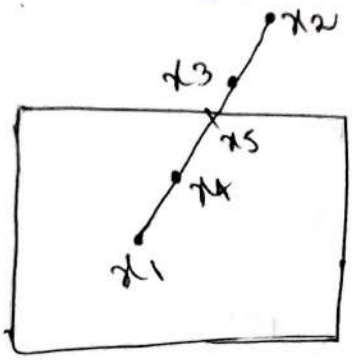
→ Midpoint Subdivision

→ line is divided into 2 parts.

→ midpoint is obtained by dividing it into two segments.

→ again division is done by finding midpoint.

→ The process is continued until line of visible and invisible category is found.




$$x_3 = \frac{x_2 + x_1}{2}, \quad x_4 = \frac{x_3 + x_1}{2}$$

$$x_5 = \frac{x_4 + x_3}{2}$$

→ Adv.

→ It is suitable for machine in which multiplication & division is not possible.

→ It can be performed by introducing clipping divides in hardware.

Algorithm

1. Calculate the ~~position~~ regioncode of both end point of line
2. Perform OR operation
3. If OR gives 0000
   then, line is visible

   else perform AND,
   if AND $\neq$ 0000 then invisible
   else line is clipped.

4. For the line to be clipped. Find midpoints,

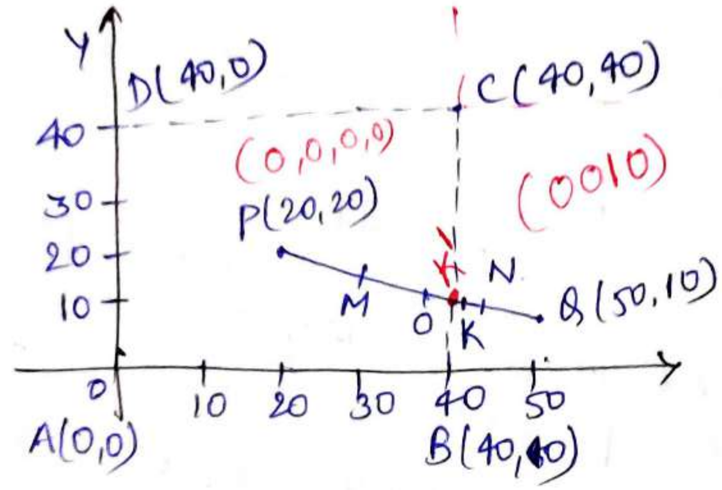
$$x_m = (x_1 + x_2)/2$$

$$y_m = (y_1 + y_2)/2$$

5. Check each midpoint, whether it is nearest to window or not.

6. If line is totally visible or totally rejected not found, repeat.

Q. Clip the line PQ against clipping window whose lower left corner is (0,0), upper left is (40,40) where P(20,20), Q(50,10)




Region of P = 0000
" " Q = 0010
Logical AND : 0000 = False

∴ PQ is clipping case.

Midpoint of PQ, $= \left( \frac{20+50}{2}, \frac{20+10}{2} \right)$

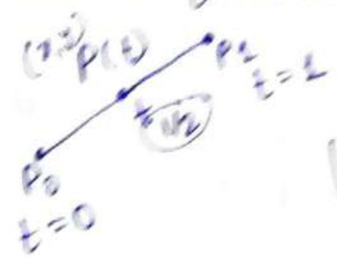
$M = (35, 15)$

PM is passed to display routine & MQ is clipping candidate line.

PQ
PM — inside line
MQ — clipped line

## Cyrus-Beck Line clipping algorithm

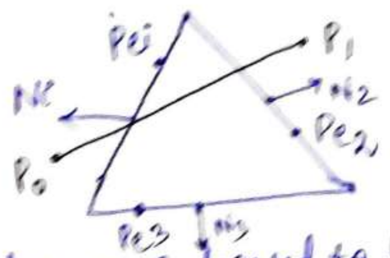
→ clipping window can be any polygon
→ concept used :-
i) dot product
ii) parametric equation line $0 \le t \le 1$ → Parameter




Parametric eqn

$$\boxed{P(t) = P_0 + t(P_1 - P_0)}$$

for $t=0$
$P(t) = P_0 + 0(\ldots)$
$= P_0$
for $t=1$
$P(t) = P_0 + 1(P_1 - P_0)$
$= P_1$




① Normal vector ⇒ Outward to line
$N_i$
$i$ = edge no.

② Take a point any where on line
$P_{ei}$ Point at edge $i$ =

Dot product

$N_i \cdot (P(t) - P_{ei})$

$< 0$ $P(t)$ inside clipping
$> 0$ $P(t)$ outside clipp..
$=$ intersection point

$$N_i \cdot (P(t) - P_{ei}) = 0$$
$$N_i \cdot (P_0 + t(P_1 - P_0) - P_{ei}) = 0$$
$$N_i P_0 + N_i \cdot t \cdot P_1 - N_i \cdot t \cdot P_0 - N_i P_{ei} = 0$$
$$N_i t P_1 - N_i \cdot t \cdot P_0 = N_i \cdot P_{ei} - N_i \cdot P_0$$
$$t \cdot N_i (P_1 - P_0) = N_i (P_{ei} - P_0)$$
$$t = \frac{N_i (P_{ei} - P_0)}{N_i (P_1 - P_0)}$$

Let $N \cdot (P_1 - P_0) = D$ [ denominator)

$D > 0$ leaving point

$D < 0$ entering point

Let $t_1, t_2$ → (entering) then select highest value of t.

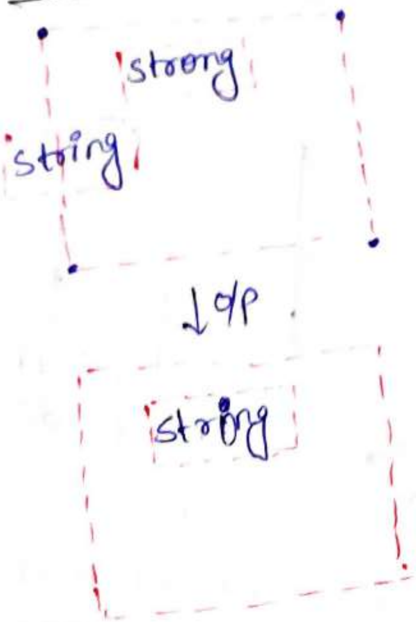
$t_3$ (leaving) smallest.

→ t value will be used in parametric eqn.

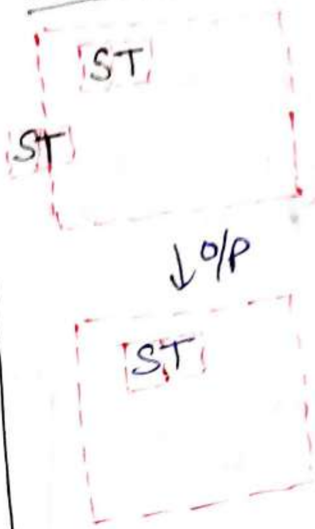
$P(t) = P_0 + t(P_1 - P_0)$

# TEXT clipping

## All or None string clipping

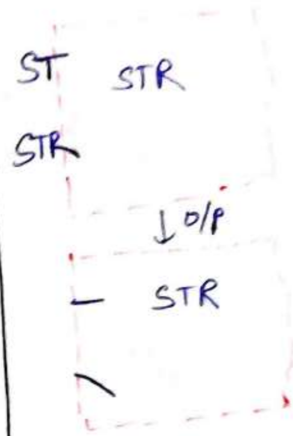



## All or None character ~~h~~clipping




## Individual character clipping

bitmap character clipping

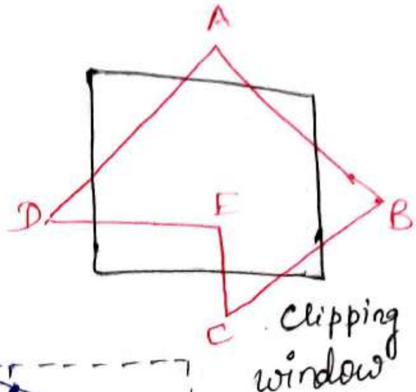



→ every character is treated individually
→ discard every character that lie on boundary or outside

## Sutherland Hodgman Polygon clipping algo:

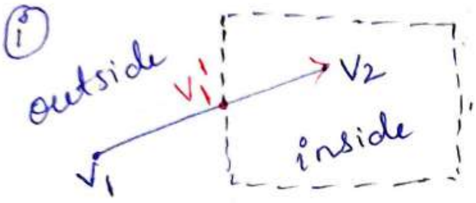
- It clips the region of the polygon lying outside the window.

→ Clip against each edge of window & obtain new set of vertices.




for finding new sequence of vertices, there are 4 cases:

(i)

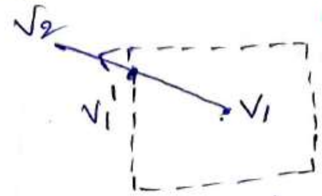



out → in

o/p → intersect point + destination point

⟶ $V_1' V_2$

(ii)

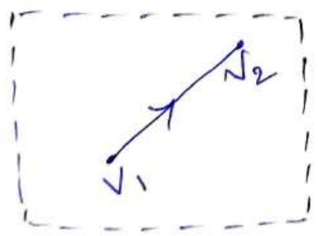



movement → (in → out)

o/p : intersection point

→ $V_1'$

(iii)

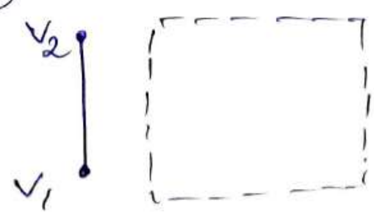



Movement → in to in

o/p → destination vertex.

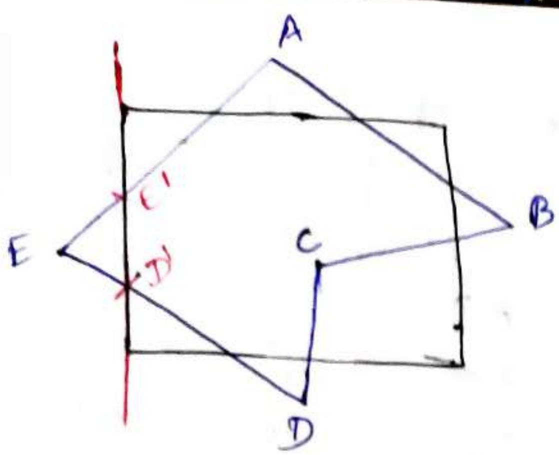
$V_2$

(iv)




movement → out to out

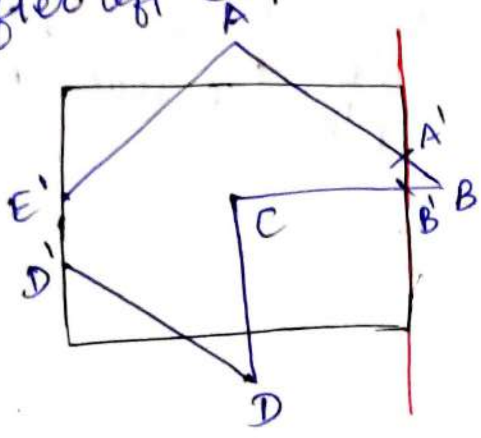
o/p → nothing/null.




① clipping against left edge

| Vertex | case | output |
|---|---|---|
| AB | in→in | B |
| BC | in→in | C |
| CD | in→in | D |
| DE | in→out | D' |
| EA | out→in | E'A |

after left clip

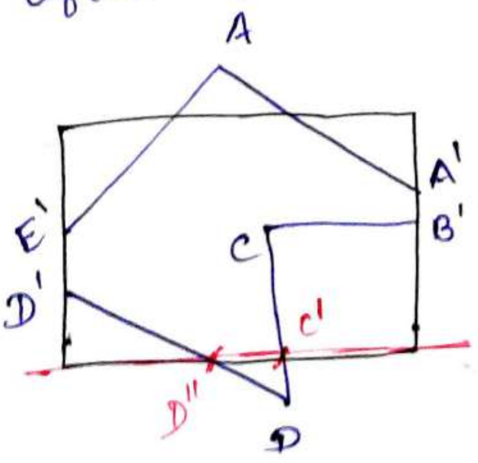



↓

② against right edge of window

| Vertex | case | o/p |
|---|---|---|
| AB | in→out | A' |
| BC | out→in | B'C |
| CD | in→in | D |
| DD' | in→in | D' |
| D'E' | in→in | E' |
| E'A | in→in | A |

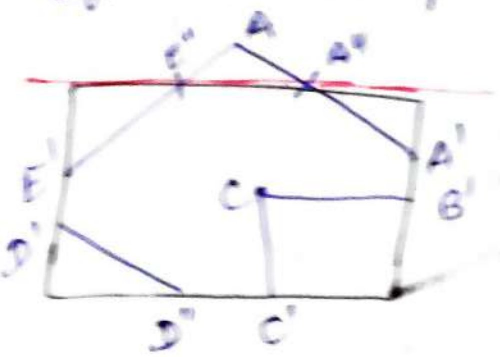
after right clip.



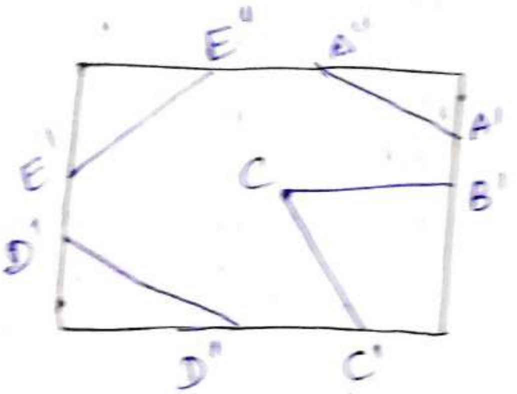

③ Bottom clipping

| Vertex | case | o/p |
|---|---|---|
| AA' | in→in | A' |
| A'B' | in→in | B' |
| B'C | in→in | C |
| CD | in→out | C' |
| DD' | out→in | D''D' |
| D'E' | in→in | E' |
| E'A | in→in | A |

after bottom clip.




after top clipping.




④ Top clipping

| Vertex | case | o/p |
|---|---|---|
| AA' | | |
| A'B' | out → in | A''A' |
| B'C | in → in | B' |
| CC' | " | C |
| C'D'' | " | C' |
| D''D' | " | D'' |
| D'E' | " | D' |
| E'A | in → out | E' |
| | | E'' |